Parametric Optimal Design Of Uncertain Dynamical Systems

Joseph T. Hays

Dissertation submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Mechanical Engineering

Adrian Sandu, Co-Chair
Corina Sandu, Co-Chair
Dennis W. Hong, Co-Chair
Shane D. Ross
Steve C. Southward

August 25, 2011
Blacksburg, Virginia

Keywords: Motion Planning, Trajectory Planning, Optimal Control, Robust Design Optimization (RDO), Dynamic Optimization, Nonlinear Programming (NLP), Multi-Objective Optimization (MOO), Uncertainty Quantification, Generalized Polynomial Chaos (gPC), Collocation, Uncertainty Apportionment, Tolerance Allocation, Multibody Dynamics, Differential Algebraic Equations (DAEs), Ordinary Differential Equations (ODEs)

Parametric Optimal Design Of Uncertain Dynamical Systems

Joseph T. Hays

Abstract

This research effort develops a comprehensive computational framework to support the parametric optimal design of uncertain dynamical systems. Uncertainty comes from various sources, such as: system parameters, initial conditions, sensor and actuator noise, and external forcing. Treatment of uncertainty in design is of paramount practical importance because all real-life systems are affected by it; not accounting for uncertainty may result in poor robustness, sub-optimal performance and higher manufacturing costs.

Contemporary methods for the quantification of uncertainty in dynamical systems are computationally intensive which, so far, have made a robust design optimization methodology prohibitive. Some existing algorithms address uncertainty in sensors and actuators during an optimal design; however, a comprehensive design framework that can treat all kinds of uncertainty with diverse distribution characteristics in a unified way is currently unavailable. The computational framework uses Generalized Polynomial Chaos methodology to quantify the effects of various sources of uncertainty found in dynamical systems; a Least-Squares Collocation Method is used to solve the corresponding uncertain differential equations. This technique is significantly faster computationally than traditional sampling methods and makes the construction of a parametric optimal design framework for uncertain systems feasible.

The novel framework allows to directly treat uncertainty in the parametric optimal design process. Specifically, the following design problems are addressed: motion planning of fully-actuated and under-actuated systems; multi-objective robust design optimization; and optimal uncertainty apportionment concurrently with robust design optimization. The framework

advances the state-of-the-art and enables engineers to produce more robust and optimally performing designs at an optimal manufacturing cost.

Acknowledgements

I'd like to thank the many people who have positively contributed to my experiences at Virginia Tech as I have pursued these research investigations.

First, I'd like to thank my many lab-mates from the Robotics & Mechanisms Lab (RoMeLa), the Advanced Vehicle Dynamics Lab (AVDL), and the Computational Science Lab (CSL) for the many laughs, stimulating discussions, and friendship. I will fail in naming them all but most of them are: Karl Muecke, Ivette Morezzani, Gabriel Goldman, Derek Lahr, Eric Williams, Jeakweon (JK) Han, Mike Hopkins, Ping Ren, Viktor Orekhov, Seungmoon Song, J. Blake Jeans, Paul D'Angio, Rob Nguyen, Robert Mayo, Ryan Colby, Bryce Lee, Dr. Kyung Bok Lee, Prof. Bohee Lee, and Prof. Young-Jae Ryoo, Emmanuel (Manny) Blanchard, Anake Umsrithong, Jeremy Kolansky, Tae-Hyuk Ahn, Alexandru Cioaca, Hong Zhang, Vishwas Rao, Evgeniy Zharovsky, Mihai Alexe. Many thanks to Jacob Dodson for hours of stimulating discussion and study. Many thanks to Prof. Curtis Stern for being a great mentor.

Without questions, many thanks are in order for the staff members of the Mechanical Engineering and Computer Science Departments. Specific staff that consistently helped with a friendly smiles are: Cathy Hill, Sue Teel, Brandy McCoy, and Melissa Williams. Thanks to you all for the countless hours of cleaning up the messes, guidance, and frequent thankless service.

To my committee members, Dr. Shane Ross and Dr. Steve Southward, thank you for your contributions to my education and for your insight and feedback on my work.

To my advisors, Drs. Adrian and Corina Sandu and Dr. Dennis Hong, thank you for your guidance, patience, feedback, funding, friendship, and support.

To my parents, in-laws, and siblings, thank you for your love, prayers, and support.

To my wonderful wife and best friend, Michelyn, and to my lovely children—Taryn, Grayden, Ian, and Nathan—your sacrifice, patience, love, prayers, encouragement, confidence, and support were invaluable! I could not have done it without you! I love you all!

Finally, to my Maker, thank you for the gift of an inquisitive mind, for opening the doors-of-opportunity, and for the consistent guidance and quite assurances.

Contents

List of Figures

List of Tables

List of Variables (Nomenclature)

Independent variables	
t	Time
ω	Random event
General	
x, X	Non-bolded variables generally indicate a scalar quantity
$\boldsymbol{x}, \boldsymbol{X}$	Bolded lower case variables are vectors, upper case variables are matrices
$\vec{\boldsymbol{x}}$	Alternative vector notation
$\angle\boldsymbol{x}$	Angle of the vector $\boldsymbol{x}$
ξ	Random variable
x_i	Bottom right index *generally* indicates a state (with occasional exceptions).
$x^{\boldsymbol{j}}$	Top right index generally indicates a stochastic coefficient, or mode.
${}_{k}x$	Bottom left index generally associates x to a specific collocation point.
${}^{a}x,\ {}^{u}x$	Top left annotations indicate if a given variable is *actuated* or *unactuated*.
${}^{u}_{k}x^{j}_{i}$	A single variable showing the four major annotations
$(\ \ )^{\boldsymbol{T}}$	Transpose
$(\ \ )_{\mathbf{q}}\ ,\ \frac{\partial}{\partial q}$	Partial derivative notations
$(\ \ )^{\mathbf{-1}}, (\ \ )^{\#}$	Matrix inverse and pseudo inverse
$(\ \ )^{\circ}$	The quantity represents rotations in degrees
$\underline{x}, \overline{x}$	Lower and upper bounds on x
$E[x]$, μ_x	Expected value, or mean, of x
$Var[x]$, σ_x^2	Variance of x
$std[x]$, σ_x	Standard Deviation of x
$\inf(x),\ \sup(x)$	Infimum and supremum of x
Indexes & dimensions	
$n_d \in \mathbb{N}$	Number of degrees-of-freedom (DOF)
$n_{gc} \in \mathbb{N}$	Number of generalized coordinates, where $n_{gc} \geq n_d$, dependent on kinematic representation of rotation.
$n_s \in \mathbb{N}$	Number of states
$n_i \in \mathbb{N}$	Number of input wrenches, $\boldsymbol{\tau} \in \mathbb{R}^{n_i}$
$n_o \in \mathbb{N}$	Number of outputs, $\boldsymbol{y} \in \mathbb{R}^{n_o}$
$n_c \in \mathbb{N}$	Number of bilateral constraints
$p_o \in \mathbb{N}$	Polynomial order
$n_b \in \mathbb{N}$	Number of multidimensional basis terms
$n_{cp} \in \mathbb{N}$	Number of collocation points
$n_{sp} \in \mathbb{N}$	Number of B-Spline basis and control points
$m \in \mathbb{N}$	Number of B-Spline knots
$\mathscr{p} \in \mathbb{N}$	Spline *degree*

$n_{dim} \in \mathbb{N}$	Number of dimensions of the B-Spline (e.g., n_d or n_i)
$n_x \in \mathbb{N}$	Number of optimization variables ('manipulated variables')
$n_a \in \mathbb{N}$	Number of apportionment variables ('allocated variables'), where $n_a \leq n_p$
$n_c \in \mathbb{N}$	Number of constraint equations
$n_p \in \mathbb{N}$	Number of independent variables
$n_{dv} \in \mathbb{N}$	Number of dependent variables
Kinematics	
$\boldsymbol{\Phi} \in \mathbb{R}^{n_c}$	Bilateral constraints
$\boldsymbol{\Phi}^{\mathrm{ol}}$	Open-loop algebraic kinematic constraints
$\boldsymbol{\Phi}^{\mathrm{cl}}$	Closed-loop algebraic kinematic constraints
$\boldsymbol{\Phi}_{\mathbf{q}}, \boldsymbol{\Phi}_{\mathbf{qt}}, \boldsymbol{\Phi}_{\mathrm{tt}}$	Associated constraint jacobian partial matrices
$\mathbf{s}$	Algebraic relations describing kinematic assembly features (e.g., gaps, clearances, positions, etc…)
Dynamics	
$\boldsymbol{q} \in \mathbb{R}^{n_{gc}}$	Independent generalized coordinates
$\dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$	Rates and accelerations of generalized coordinates
$\boldsymbol{v} \in \mathbb{R}^{n_d}$	Generalized velocities
$\dot{\boldsymbol{v}}$	Generalized accelerations
$\boldsymbol{q}(\mathbf{0}) = \boldsymbol{q}_0 , \boldsymbol{v}(\mathbf{0}) = \boldsymbol{v}_0$	Initial conditions
$\boldsymbol{H} \in \mathbb{R}^{n_s \times n_s}$	Kinematic mapping matrix relating rates of generalized coordinates to generalized velocities
$\boldsymbol{\tau} \in \mathbb{R}^{n_i}$	Input wrenches
$\boldsymbol{M} \in \mathbb{R}^{n_s \times n_s}$	Square inertia matrix
$\boldsymbol{C} \in \mathbb{R}^{n_s}$	Centrifugal, gyroscopic and Coriolis terms
$\boldsymbol{N} \in \mathbb{R}^{n_s}$	Generalized gravitational and joint forces
$\boldsymbol{\mathcal{F}}$	Differential operator
$\boldsymbol{\mathcal{G}}$	Under-actuated differential operator
$\boldsymbol{y} \in \mathbb{R}^{n_o}$	System outputs
$\boldsymbol{\mathcal{O}} \in \mathbb{R}^{n_o}$	Output operator
$\boldsymbol{y}_{ref}$	Reference outputs
$\boldsymbol{\lambda} \in \mathbb{R}^{n_c}$	Lagrange multipliers
$\boldsymbol{\alpha} \in \mathbb{R}^{n_c}$	Acceleration constraints
Uncertainty Quantification	
Ω	Random event sample space
$f(\xi)$	Joint probability density function
$\boldsymbol{\theta} \in \mathbb{R}^{n_p}$	Vector of uncertain independent variables
$\boldsymbol{\rho} \in \mathbb{R}^{n_{dv}}$	Vector of uncertain dependent variables
$\psi \in \mathbb{R}^{p_o+1}$	Single dimensional basis terms
$\Psi \in \mathbb{R}^{n_b}$	Multidimensional basis terms
$H_n(x)$	Hermite polynomial basis
$L_n(x)$	Legendre polynomial basis

$\mathcal{N}(\mu,\sigma)$	Normal, or Gaussian, distribution with mean and standard deviation of $[\mu,\sigma]$
$\mathcal{U}(a,b)$	Uniform distribution with range of $[a,b]$
$\mathcal{A}$	Algebraic operator
$\mathfrak{D}$	Differential operator
${}_k\boldsymbol{\mu},\quad \boldsymbol{\mu} \in \mathbb{R}^{n_{cp}}$	Kth collocation point
${}_kX_i,\ \ X_i \in \mathbb{R}^{n_{cp}}$	Kth intermediate variable of the ith state representing expanded quantity
${}_k\boldsymbol{\mathcal{P}}_i,\ \ \boldsymbol{\mathcal{P}}_i \in \mathbb{R}^{n_{cp}}$	Kth intermediate variable of the ith uncertain dependent variable, λ_i, representing expanded quantity
$A \in \mathbb{R}^{n_b \times n_{cp}}$	Collocation matrix
Nonlinear Programming	
$\min\limits_{\mathbf{x}}$	Optimization objective through manipulation of $\mathbf{x}$
$\mathbf{x}$	List of manipulated variables
J	Scalar cost function
Ξ	Terminal condition terms
L	Lagrangian, or trajectory, cost terms
Γ	an auxiliary cost over the trajectory that is not integrated
w_i	Scalarlization weights for the individual input wrench contributions
a_i	Standard deviation scaling parameters
$\mathrm{t_f}$	Final time of trajectories
ς	*Soft* constraint penalty weight
$\boldsymbol{\mathcal{C}}$	Inequality constraints (typically bounding constraints)
$\mathbf{B}$	B-Spline curve
u	B-Spline knot list
$\beta^{i,\mathscr{p}}$	B-Spline basis terms of *degree* $\mathscr{p}$ and $i = 1 \dots n_{sp}$
$\boldsymbol{P} = \{\boldsymbol{p}^i\}$	B-Spline control points where $i = 1 \dots n_{sp}$
$\boldsymbol{P}' = \{\boldsymbol{p}'^i\}$	Derived control points for velocity B-Splines where $i = 1 \dots n_{sp}$
$\boldsymbol{P}'' = \{\boldsymbol{p}''^i\}$	Derived control points for acceleration B-Splines where $i = 1 \dots n_{sp}$
$\boldsymbol{\mathfrak{D}}_{i,j}$	A signed minimum distance between two geometric bodies *i* and *j*

1 Introduction

1.1 Motivation

Design engineers cannot quantify exactly every aspect of a given system. These uncertainties frequently create difficulties in accomplishing design goals and can lead to poor robustness and suboptimal performance. Tools that facilitate the analysis and characterization of the effects of uncertainties enable designers to develop more robustly performing systems and to manufacture them at lower costs. The need to analyze the effects of uncertainty is particularly acute when designing dynamical systems. Frequently, engineers do not account for various uncertainties in their design in order to save time and to reduce development costs. However, this simply delays, or hides, the cost which is inevitably incurred down-stream in the design and manufacturing processes. Ultimately, if a robust system design is to be achieved, uncertainties must be accounted for up-front during the design process.

1.2 Research Contributions

The inclusion of uncertainty in design is of paramount practical importance because all real-life systems are affected by it. Uncertainty in dynamical systems comes from various sources, such as: system parameters, initial conditions, sensor and actuator noise, and external forcing. Designs that ignore uncertainty often lead to poor robustness, suboptimal performance, and higher build costs.

Contemporary methods for the quantification of uncertainty in dynamical systems are computationally intensive and have made a robust optimal design methodology prohibitive. Currently, there is no existing comprehensive framework that can treat all kinds of uncertainty with diverse distribution characteristics in a unified way during a dynamic optimization.

Therefore, this research effort develops a novel computational framework for the parametric optimal design of uncertain dynamical systems. This computational framework resides at the intersection of the following subjects: multibody dynamics, uncertainty quantification, and nonlinear optimization (see Figure 1.1). The approach employs the relatively new Generalized Polynomial Chaos (gPC) methodology to quantify various sources of uncertainty found in dynamical systems, and to solve the resulting uncertain differential equations (UDEs) with a Least-Squares Collocation Method (LSCM). This technique is significantly faster computationally than traditional sampling methods and makes the treatment of uncertainty during dynamic optimization feasible. As such, new design questions related to uncertain dynamical systems can now be answered through this new framework.

Specifically, the following use cases are addressed:

- **Motion planning of Uncertain Systems:** Direct treatment of system uncertainties during the optimal motion planning process enables designers, for the first time, to produce robust optimally performing motion plans for the entire family of realizable systems that satisfy actuator and collision avoidance constraints. The framework is developed for fully-actuated and under-actuated uncertain dynamical ODE systems through forward, inverse, and hybrid dynamics formulations.
- **Parametric robust design optimization (RDO):** Direct treatment of system uncertainties during RDO enables designers, for the first time, to characterize the Pareto optimal trade-off surface for an entire family of realizable systems; this yields a shifted, or off-set, Pareto optimal trade-off surface—when compared to contemporary deterministic RDO approaches—that more accurately characterizes the optimal designs for the entire probability space. The new framework uses a constraint-based multi-objective optimization (cMOO) formulation; problems that are nonlinear, have active constraints, or opposing design objectives, are shown to benefit from this new framework.
- **Optimal uncertainty apportionment (OUA):** For the first time, the framework gives design engineers the ability to treat both geometric and non-geometric uncertainties of relatively large magnitude during a concurrent RDO/OUA design process. Performance objectives are redefined as constraints; this yields a cMOO formulation where the problem objective is a weighted function of the various manufacturing cost-uncertainty trade-off curves. Subsequently, robust optimally performing systems are determined at an optimal manufacturing cost.

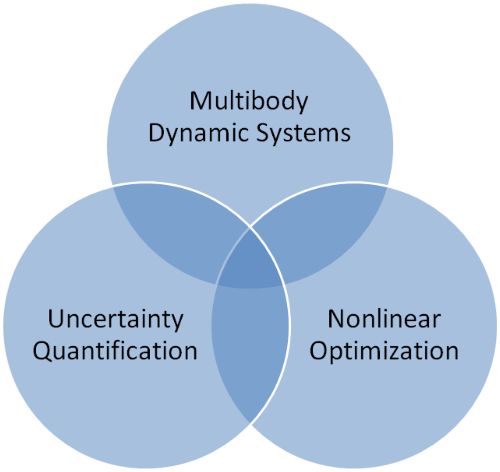

Figure 1.1—Elements of the computational frameworks for the optimal design of uncertain dynamical systems include: multibody dynamics, uncertainty quantification, and nonlinear optimization.

1.3 Literature Review

In the following, a review of the literature is presented where works specifically related to uncertainty quantification, motion planning, robust design optimization, and uncertainty allocation are covered.

1.3.1 Uncertainty Quantification:

1.3.1.1 Monte Carlo Methods

The Monte Carlo (MC) method is considered the most robust method of uncertainty quantification. The method is quite simple; the probability space of the system is randomly sampled n times and statistical measures are determined from the ensemble [1]. MC provides a consistent error convergence rate independent of the number of uncertainties. However, the convergence rate of $1/\sqrt{n}$ is relatively slow.

Alternatively, quasi-Monte Carlo (QMC) methods deterministically sample the probability space with *low-discrepancy sequences* (LDS). QMC is reported to show improved constant convergence, $(\log n)^d/n$, for relatively low dimensioned problems when compared to MC [2, 3]; where d is the number of dimensions.

1.3.1.2 Generalized Polynomial Chaos

Generalized Polynomial Chaos (gPC) is a relatively new method that is rapidly being accepted in diverse applications. It's origins come from early work by Wiener in the 1930's where he introduced the idea of *homogenous chaoses* [4]. His work made use of Gaussian distributions and the Hermite orthogonal polynomials. Xiu and Karniadakis generalized the concept by expanding the list of supported probability distributions and associated orthogonal polynomials [5, 6]; where the Galerkin Projection Method (GPM) was initially used. In [6-8], Xiu showed an initial collocation method based on Lagrange interpolation. A number of Collocation point selection methods were also show including tensor products and Smolyak sparse grids.

In [9], Sandu et. al. introduced the least-squares collocation method (LSCM) and used the roots of the associated orthogonal polynomials in selecting the sampling points. Cheng and Sandu showed the LSCM maintains the exponential convergence of GPM yet was superior in computational speed in [10]; where the Hammersley LDS data set was the preferred method in selecting collocation points. This work also presented a modified time stepping mechanism where an approximate Jacobian was used when solving stiff systems.

1.3.1.3 Multi-Element gPC

The accuracy of gPC deteriorates over time in long simulations and is dependent on continuity of the system. In an effort to address these two concerns, Wan and Karniadakis developed multi-element gPC (MEgPC) [11, 12]. This method discretizes the probability space into non-overlapping partitions. Within each partition the traditional single element gPC is performed. Summing element integrations provides a complete integration of the full probability space. The algorithm presented adaptively partitioned the space based on estimates of error convergence. When an error estimate deteriorated to a specified point the element was split. The initial work was developed for the GPM methodology using uniform distributions. MEgPC was subsequently extended to arbitrary distributions in [13, 14]. Foo developed a collocation-based MEgPC in [15] and further extended the method to support higher dimensions using ANOVA methods in [16].

As an alternative to MEgPC, Witteveen and Iaccarino developed a similar multi-element method based on gPC called the simplex elements stochastic collocation (SESC) method. This method adaptively partitions the probability space using simplex elements coupled with Newton-Cotes quadrature. Their method has show an O(n) convergence as long as the approximating polynomial order is increased with the number of uncertainties.

1.3.1.4 Recent Applications of gPC/MEgPC

The origins of gPC come from thermal/fluid applications; however, its adoption in other areas continues to expand. Sandu and coworkers introduced its application to multibody dynamical systems in [9, 17]. Significant work has been done applying it as a foundational element in parameter [8, 18-35] and state estimation [34, 36], as well as system identification [37]. Relatively recent work has applied gPC to both classical and optimal control system design [25, 38, 39]. Also, MEgPC has been used applied to uncertainty quantification in power systems [40] and mobile robots [41].

1.3.2 Motion Planning and Optimal Control

1.3.2.1 Deterministic Kinodynamic Motion Planning

In [42], Park presents a nonlinear programming approach to motion planning for robotic manipulator arms described by deterministic ODEs. The main contribution of Park's work is to define new cost terms that capture actuator force limiting characteristics; where actuator velocities and resulting feasible torques are defined. Park's formulation utilizes quintic B-Splines to provide a tractable finite dimensional search space along with Quasi-Newton based solver methods (e.g., BFGS). Additionally, he approaches obstacle avoidance by defining distance constraints with the growth function technique from [43].

Sohl, Martin, and Bobrow presented a series of papers that document their excellent work in the area of optimal manipulator motions. At the heart of their work is the use of a novel geometric formulation of robot dynamics based on the differential geometry principles of Lie Groups and Lie Algebras [44-46]. The approach provides a few critical properties that streamline the optimal motion planning problem; first, the geometric dynamics formulation has an equivalent recursive formulation that provides O(n) computational complexity; second, use of the Product-of-Exponentials (POE) in the formulation provides a straight-forward approach to calculating the gradient of the optimal motion planning objective function. Access to an exact analytic gradient improves the nonlinear programming solution by helping avoid premature convergence or excessive searching for the frequently ill-conditioned motion planning problems. In [47], Martin and Bobrow present a minimum effort formulation for open chain manipulators based on the recursive geometric dynamics. A detailed presentation for the recursive calculation of the analytic objective function gradient is a major contribution of this work. They also use cubic B-Splines to provide a finite dimensional search space. In [48], Sohl and Bobrow extend the work to address branched kinematic chains; in [49-51] they again extend the work to address under-actuated manipulators; and in [52, 53] the methods are applied to the specific design problem of maximizing the weightlifting capabilities of a Puma 762 Robot. Throughout this series of work the sequential quadratic programming (SQP) technique is used for the constrained optimization; however, in [54], a Newton type optimization algorithm is developed that reuses the analytic gradient and Hessian information from the geometric dynamics. In [55], Bobrow, Park, and Sideris, further extend the work to solve infinite-dimensional problems using a sequence of linear-quadratic optimal control sub-problems and cover minimum energy, control effort, jerk and time. Finally, reference [56] extend the geometric-based optimization methods to more general dynamic systems including those with closed-kinematic loops and redundant actuators and sensors.

Another inspiring body of research comes from Xiang, Abdel-Malek et. al. [57-64] where analytic derivatives for the optimization cost of general open, branched, and closed looped systems, described by recursive Lagrangian Dynamics, is presented. Formulations are based on the Denavit-Hartenberg kinematic methods, cubic B-Splines, and SQP-based solvers. Application emphasis focuses on the motion planning of over-actuated 3D human figures; where models with as many as 23 DOFs and 54 actuators are used to design natural cyclic walking gaits. A combination of inverse and forward dynamics formulations are used, however, their formulation avoids explicit numerical integration (required in a sequential nonlinear programming (SeqNLP) methodology). Instead, their formulation makes use of the simultaneous nonlinear programming (SimNLP) methodology; which discretizes the EOMs over the trajectory of the system and treats the complete set of equations as equality constraints for the NLP. Therefore, the SimNLP has a much larger set of constraints than the SeqNLP approach, but, enjoys a more structured NLP that typically experiences faster convergence. (Note: the definitions of SimNLP and SeqNLP come from [65, 66].) Additional contributions of Xiang's work include human walking specific constraint formulations.

In [67], Park and Park present a convex motion planning algorithm that determines a stable motion plan that approximates a reference motion plan for a humanoid robotic system. The use case stems from applying measured joint trajectories from a human and applying them to a humanoid robot; this generally results in an unstable reference trajectory for the robot. However, Park and Park present a second-order cone formulated motion planning problem that determines a stable motion plan yet still approximates the reference trajectory in a least-squares sense. Similar work was presented in [68], where reference motion plans are refined online through use of a recursive forward dynamics based optimization framework with analytic derivatives. The resulting motion plan is determined in the joint space versus the wrench space.

Lim et. al. present an interesting extension to the optimal motion planning problem in [69], where motion primitives are extracted from an ensemble of optimal motions determined through repeated optimizations of a perturbed walking surface. The technique is applied to the novel tripedal robot STriDER. The primitives are determined by extracting principle components from the ensemble of optimal motion plans over varying heights of the walking surface. Once determined, the motion primitives provide a fast reference motion plan for online use. Unlike the previously referenced papers, Lim's work used Power Series to parameterize the infinite search space. The design sought for a minimum effort gait. Hays et al. have investigated the co-design of STriDER's motion plan and mechanical properties in [70].

1.3.2.2 Sample-Based Motion Planning

The term *stochastic motion planning* has multiple meanings in the literature. One definition is related to motion planning techniques that randomly sample a design space in search of a feasible motion plan. Some predominant examples of these techniques include: Rapidly-exploring Random Trees (RRTs) [71, 72], probabilistic Roadmaps (PRMs) [71, 72], and the relatively new Rapidly-exploring Random Graph (RRGs) [73]. A second definition deals with motion planning of a system that is stochastic, or uncertain. This work equates *stochastic motion planning* with the second definition; motion planning for an uncertain system. The first definition will subsequently be referred to as *sample-based motion planning*.

1.3.3 Motion Planning of Uncertain Systems

Very little research has been performed in the area of stochastic motion planning. LaValle treats sensor uncertainty with RRTs in [71]. Barraquand addresses both actuator and sensor uncertainty in a stochastic dynamic programming (DP) framework but this work only addresses the kinematics of the system [74]. Park also presents a kinematic only motion planning solution for systems with sensor and actuator uncertainties based on the Fokker-Planck equation [75]. Erdmann's early work on the *back-projection* method also only addressed sensor and actuator noise and was limited to first-order dynamic models [76].

In [77], Kewlani presents an RRT planner for mobility of robotic systems based on gPC but refers to it as a stochastic response surface method (SRSM). This technique is similar in spirit to the work presented in this work. The main difference is Kewlani's solution is developed only for determining a feasible motion plan. This work provides an optimal design, not a feasible design, framework including, but not limited to, motion planning [78-81].

1.3.3.1 Optimal Control of Dynamic Systems With Unilateral Constraints

With the exception of the work from Xiang et. al., the literature reviewed in Section 1.3.2.1 primarily treated smooth dynamical systems. The presence of discontinuities in an optimization setting complicates the problem significantly. There is little literature related to the optimization of discontinuous systems. Stewart and Anitescu investigated the optimal control of discontinuous differential equations [82] and found that even when a solution only crosses a discontinuity and does not stay on it the gradients have errors proportional to the step size of the solver. The use of a mechanical system with Coulomb friction is used as an illustration and guidelines for proper *smoothing* are provided as functions of the solver's step size.

Zefran and Kumar investigated the optimal control of dynamic systems under the influence of unilateral constraints involving contact (e.g. walking, running, and manipulation). Their work presents a method for solving a constrained optimal control problem as an unconstrained calculus of variations problem that is solved using an integral formulation [83].

In [84], Tassa and Todorov present an interesting solution to the optimal control of dynamic systems with unilateral contact constraints. Their approach propogates sensor uncertainty through the EOMs and then a stochastic linear complementarity problem (SLCP) is formulated that results in continuously differentiable states. In essence, their SLCP approach stochastically smooths the discontinuity sufficiently for the optimal control problem to proceed.

While non-gradient-based optimization methods are an option for discontinuous systems, they are significantly less efficient where excessive system evaluations are required. This proposal, therefore, focuses on developing a computational framework for the optimal design of multibody dynamic systems solved with gradient-based methods.

1.3.3.2 Optimal Co-Design of Dynamic Systems

One of the use cases the novel computational frameworks may address is the optimal design of a system's controller and mechanics simultaneously. Reference [70] presents investigative results into this problem in a deterministic setting for the novel tripedal robot, STriDER.

In [85], Fathy used first-order optimality conditions to shows that the true optimum can only be found by either a simultaneous or bi-level formulation; meaning, the optimization of the controller must be performed simultaneously/bi-level with the optimization of the mechanics. However, the sequential and iterative formulations result in sub-optimal solutions. Similar work by Alyaqout in [86, 87] analyzes the coupling between the mechanics design and the robust control optimization problems. Parameter sensitivities are analyzed between the coupling and robustness of the system and shown on a vehicle suspension problem.

Ravichandran uses a simultaneous optimal co-design approach for the design of a two-link manipulator's nonlinear PD controller and its gravity balancing link. Results confirm the co-design approach yields a superior design than individual optimizations of the respective subsystems [88].

1.3.4 Parametric Robust Design Optimization

Many different formulations and solver techniques are presented in the literature for approaching system design through dynamic optimization. The selected case-study to showcase the new design framework of this work is related to the optimal design of vehicle suspension systems; therefore, particular preference is given to the review of related literature.

A common theme is found in most of the vehicle suspension optimal design related works, namely, proper design of a suspension needs to address opposing design requirements related to passenger comfort (*ride*), suspension travel (*rattle*), and tire/road holding forces (*holding*). This necessitates a multi-objective approach to the optimal design problem. For example, [89-95] all employed linear vehicle models to construct a Pareto trade-off surface between the three referenced objectives.

Various methods were used to model the road input. A number of authors used stationary ergodic Gaussian inputs for linear models and used a power-spectral density (PSD) transformation of the system's linear frequency response [89, 90, 92, 95]. Additional attention was given to frequency weighted power-spectrum inputs based on standards such as ISO 2631 [90, 92, 95, 96]. This approach directly accounts for uncertainty in the road input of a linear system. Verros used the same Gaussian uncertain inputs for nonlinear quarter-car models through application of a Monte Carlo sampling technique [90]. These are examples of continuous road irregularity inputs. Additional authors treated isolated road irregularities such as speed bumps and potholes [96, 97].

Work related to active and semi-active suspension designs in addition to passive designs were presented in [91, 96-98]. The excellent work presented by Jazar [94] and by Gobbi and co-workers [89] approached the problem analytically; where a majority of the literature used numerical techniques showing a slight preference to DS-based (such as Genetic Algorithms [91, 93, 96]) versus NLP-based formulations [91, 97]. Also, adjacent applications related to rail [92, 99, 100] and bicycle [101, 102] optimal suspension designs were approached in a similar fashion.

In [103, 104], Hays et. al. present the robust design optimization framework found in this work.

1.3.5 Optimal Uncertainty Apportionment / Tolerance Analysis and Allocation

The design for manufacturability community has long studied the adverse effects of uncertainty in kinematically assembled mechanical systems; these studies are generally referred to as *tolerance analysis* and *tolerance allocation*. *Tolerance analysis* approaches the problem from the perspective of analyzing uncertainty accumulation in assembled systems. Conversely, *tolerance allocation* determines the best distribution and maximized magnitudes of uncertainties such that the final assembled system satisfies specified assembly constraints. Initially, these studies only treated rigid kinematic relations of the dimensional uncertainties [105-108], but have been extended to include flexible [109] and dynamical systems [110-112].

Early works by Chase and co-workers investigated various techniques for the allocation of geometric tolerances subject to cost-tolerance tradeoff curves. Their approaches included both nonlinear programming (NLP) and analytic Lagrange multiplier solutions for two-dimension (2D) and three-dimension (3D) problems where a number of cost-tolerance models were analyzed and compared [105, 106]. Extensions of their initial work included the costs of different manufacturing processes for a given feature; thus, an optimal cost-to-build included the optimal set of processes to complete the production of the various components [107].

The conventional design methodology prescribed a sequential approach where optimal dimensions were designed first followed by a *tolerance analysis* or *tolerance allocation* [110]. However, researchers quickly learned that superior designs could be achieved by designing the optimal system parameters concurrently with the allocation of the geometric tolerances [111, 113].

In [114], Hays et. al. present the optimal uncertainty apportionment framework found in this work.

1.4 Outline of Dissertation

The motivation, research contributions, and review of literature for this research effort have been presented. A brief review of multibody dynamics (Section 2) and numerical optimization (Section 3) are given to establish context for the scoped problems the research addressed. Section 4 presents a general overview of the gPC approach to uncertainty quantification used in the work. Section 5 presents the new motion planning framework for uncertain multibody dynamical systems where fully-actuated and under-actuated kinodynamic systems are addressed. Section 6 presents the new parametric robust design optimization (RDO) framework with an emphasis in Pareto optimal trade-off characterization through a constrained multi-objective formulation. Section 7 presents the new framework for optimal uncertainty apportionment (OUA) that is performed concurrently with RDO. Finally, Section 8 concludes the results of this research effort and discusses future topics that may be considered for investigation.

2 Dynamical Systems

2.1 Motivation

This chapter presents various classifications of dynamical systems followed by a very brief review of some contemporary methods to model and solve these systems. Classifications discussed include: smoothness, certainty, actuation schemes, kinematic constraints, topology, and the common types and formulations of equations of motion. The chapter concludes with a description of the ordinary differential equation (ODE) and differential algebraic equation (DAE) formulations used in this work. The purpose in reviewing this material is to clearly define the types of dynamical systems the new optimal design frameworks address. Specifically, the computational frameworks treats uncertain smooth bilaterally constrained dynamical systems of varying topologies and actuation schemes.

The interested reader is referred to [115-120] for a more comprehensive introduction to the topics presented in this chapter.

2.2 Smoothness and Certainty

Two ways of classifying a dynamical system are, (1) by its smoothness, and (2) by its determinism (illustrated in Figure 2.1).

The smoothness of the systems considered here is characterized by *Lipschitz continuity*; which simply limits how fast the difference between two functional values of any two points can change. If a system's *Lipschitz constant* is bounded then the system can be classified as *continuous*; otherwise, the system is classified as *discontinuous*. Classic examples of discontinuous systems are those experiencing dry friction, or impact.

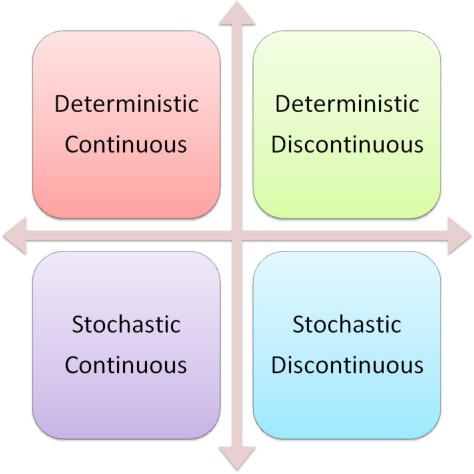

Figure 2.1—Smoothness and determinism are two dimensions in classifying a dynamical system. Lipschitz continuous systems have bounded rates of change. Examples of discontinuities in multibody dynamic systems include impact and dry friction.

If a given system model specifies all attributes as known quantities then the model is classified as *deterministic*. Otherwise, the presence of any uncertain system attributes—and the uncertainty is accounted for in the model through some mechanism—results in a *stochastic* model.

2.3 Actuation Schemes

Dynamical systems are also diversely actuated and can be classified as: passive, fully-actuated, under-actuated, over-actuated, and under-over-actuated (see Figure 2.2).

Passive systems do not have associated actuators, they simply respond to external forces (e.g. gravity and disturbances). A traditional car suspension is an example of a passive dynamic system. A *fully-actuated* system has one actuator for each of the system's degrees-of-freedom. An industrial robotic arm is a fully-actuated system. *Under-actuated* systems have both passive and actuated degrees-of-freedom (DOF), however, any actuated DOF only has one associated actuator. A humanoid robot is an example of an under-actuated system. An *over-actuated* system is fully-actuated but has one or more DOFs with more than one actuator. Over-actuated systems are synonymously referred to as *redundantly-actuated* systems in the literature. Many joints in animals are redundantly actuated. Think of a human shoulder; there are more than three muscles actuating its 3 DOFs. An interesting characteristic of over-actuated DOFs is that the associated limits of operation are influenced by the presence of multiple actuators. Again, think of a human's joints, most joints are limited in their range of motion by the various muscles attached to it. This makes the analysis and control of over-actuated systems more challenging.

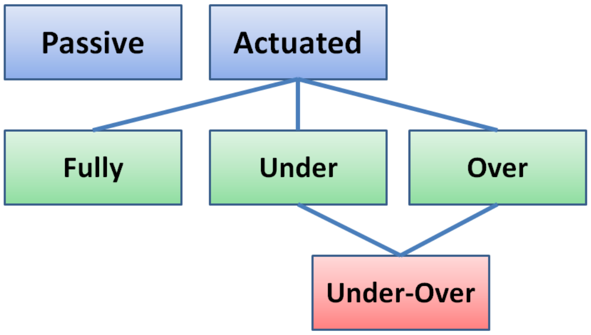

Figure 2.2—Dynamical systems can also be classified by their actuation schemes. A passive system has no actuators. An actuated system can be fully-actuated, under-actuated, over-actuated, or under-over-actuated.

Finally, *under-over-actuated* systems have at least one passive, and one over-actuated DOF. Humans and most animals have redundantly actuated joints, however, the DOFs of their base are passive. Therefore, they can be classified as an under-over-actuated system. Systems such as these are occasionally referred to as having a *floating base*.

2.4 Constrained Dynamic Systems

Many physical systems operate under kinematic constraints. There are a number of different kinds of kinematic constraints. A very brief layman's description of the common constraints found in multibody dynamical systems is presented. Figure 2.3, borrowed from [121], shows the relationships of these constraint groups. Again, the purpose of this review is to establish context of what problems this work directly addresses and what it does not.

2.4.1 Bilateral Constraints:

Bilateral constraints establish a dependency relationship in the coordinates. Meaning, the presence of a bilateral constraint forces two or more coordinates to maintain an equality relation between them. There are two main classes of bilateral constraints: *holonomic*, and *nonholonomic* constraints.

Holonomic constraints are a very common type of constraint found in engineered physical systems. Simply stated, a system that is constrained only by holonomic constraints has an equal number of controllable DOFs to the total DOFs of the system. If a holonomic system is described by *independent* generalized coordinates then the number of coordinates is equal to the number of DOFs. Holonomic constraints that are explicit functions of time are known as *rheonomic* constraints; without having an explicit dependence on time they are known as *scleronomic* constraints. Also, holonomic constraints are typically functions of positional states.

Unlike holonomic constraints, *nonholonomic constraints* yield a system that has less controlled DOFs than the total DOFs of the system. In other words, any given state of the system is path dependent. The most intuitive example is an automobile. The automobile can move anywhere on a two dimensional plane and have any orientation so it has 3 DOFs, however, the system does not have 3 actuated DOFs. Nonholonomic constraints are typically functions of velocity states.

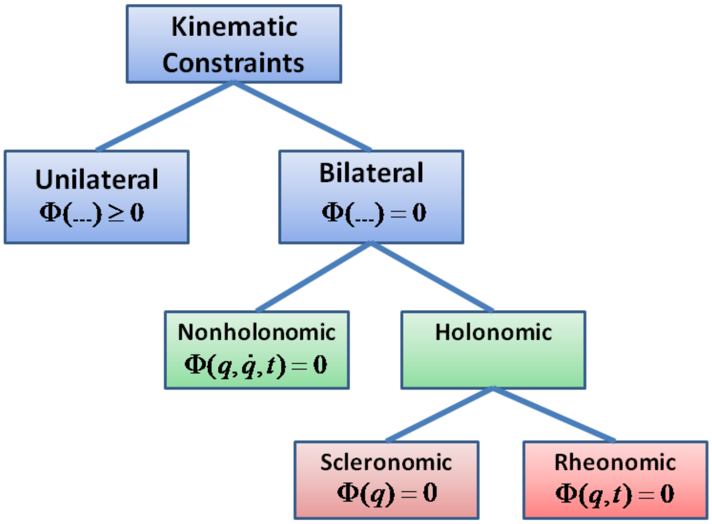
Kinematic Constraints
Unilateral $\Phi(...) \geq 0$
Bilateral $\Phi(...) = 0$
Nonholonomic $\Phi(q, \dot{q}, t) = 0$
Holonomic
Scleronomic $\Phi(q) = 0$
Rheonomic $\Phi(q, t) = 0$

Figure 2.3—Dynamic systems can be constrained by unilateral and bilateral constraints. Holonomic constraints are integrable and typically involve the generalized coordinates. Nonholonomic constraints are not integrable and typically involve the derivatives of the generalized coordinates.

2.4.2 Unilateral Constraints

Unilateral constraints are similar to bilateral constraints in that they establish a dependency relationship in the coordinates of the system, however, this type of relationship is not an equality, but an inequality. Therefore, if the system configuration is not on the boundary of the constraint, then the system behaves as if the constraint didn’t exist at all. The most intuitive example of a unilateral constraint is that of geometric contact. Two bodies have a unilateral constraint between them; as long as they are not in contact with each other they can move as if there was no constraint. However, if they come into contact suddenly the constraint becomes active and induces appropriate forces on the two systems to satisfy the constraint.

Some authors define, or classify, unilateral constraints as nonholonomic constraints. Additionally, multibody dynamical systems subject to unilateral contact constraints require specialized numerical time-stepping formulations and collision detection algorithms; the interested reader is encouraged to consult [82, 122-145]. Also, unilateral constraints are discontinuous in nature. This causes difficulties in gradient-based optimization searches and is out of the scope of this research effort.

2.4.3 Topology

Constrained multibody dynamical system can be classified in three primary topological groups (listed in increasing order of complexity):

- Open kinematic chains (a.k.a. serial manipulators)
- Branched kinematic trees
- Closed kinematic loops (a.k.a. parallel manipulators)

This classification is mentioned simply to mature the setting of this research proposal. Various dynamics formulations exist; each providing differing advantages and limiting assumptions. Different topologies may or may not be supported by a given formulation.

In the following, a review of the dynamics formulations used for the work's case studies is presented, however, it is important to mention that the stochastic optimal design framework that this work provides is independent of which dynamics formulation is selected for developing the equations of motion.

2.5 ODEs, DAEs, and DAIs

There are three main types of equations that model multibody dynamical systems:

- Differential Equations (DEs / ODEs)
- Differential Algebraic Equations (DAEs)
- Differential Algebraic Inequalities (DAIs)

Scoping the presentation to lumped parameter systems simplifies DEs to ordinary differential equations (ODEs). Any constraints in a system described by ODEs are implicitly embodied in the ODEs.

DAEs on the other hand are a combination of ODEs and algebraic equations (AEs). The AEs explicitly describe bilateral constraints the ODEs must conform to for all time. DAEs are more difficult to solve numerically than ODEs and typically require some form of constraint stabilization calculation. Optimal design of uncertain DAEs will not be addressed in this research but is recommended for consideration in future research.

DAIs are the most versatile of the three. They can provide a richer model of the physical world—where the interactions between a dynamic system and its environment are captured. These environmental interactions are embodied in explicit unilateral constraints described with inequalities. Inequalities of this type result in discontinuous effects—such as impact, contact, and joint limits—therefore, DAIs are of the discontinuous class of dynamics discussed in Section 2.2.

DAIs are the most expensive computationally and require specialized solvers [82, 122-145]. Also, their discontinuities pose difficult mathematical and algorithmic challenges for the optimization of such systems. As such, the optimization of uncertain DAI-based systems is out of the scope of this research and is suggested for future investigations.

2.6 Common EOM Formulations

There are many methods to formulate the equations of motion (EOMs) for a dynamical system. All adhere to Newton's laws but each offers different advantages and limitations. The more common formulations include:

- Newton/Euler Equations [115-117, 119]
- Euler/Lagrange Equations [115-117, 119]
- Hamilton's Equations [115-117, 119]
- Linear/Bond Graphs [146]
- Geometric Methods [44, 45, 147]

All of these methods are capable of constructing ODEs, DAEs. Again, the parametric optimal design frameworks developed are independent of which dynamics formulation is selected for developing the equations of motion.

2.6.1 *O(n)* Formulations

The numerical solution of traditional constrained multibody dynamic system formulations have a computational complexity of *$O(n^3)$*. The cost to solve such systems can grow rapidly with the number of states in the worst-case. However, in response to the insatiable demands for *more speed*, new formulations have been developed in the last few decades that provide *O(n)* computational complexity [46, 48, 141-144, 148-150]. Therefore, the computational cost only increases linearly with the number of system states which results in dramatically faster solvers. Featherstone has develop the fastest documented rigid-body dynamics algorithm that provides *O(log(n))* complexity if *n* parallel processors are available [121, 123, 151-154]. However, each method comes with limitations as to what types of systems it can support (e.g. most are limited to open and tree topologies, most do not support nonholonomic constraints).

Another important point to make regarding the recursive *O(n)* and *O(log(n))* methods, they drastically simplify the modeling process. Traditional methods are very manually and symbolically intensive. However, this new class of recursive methods only requires a few lists and joint libraries to automatically construct the recursive model. For example, if a list of bodies, their respective inertial and geometric properties, and connectivity graph/tree are provided, then the recursive model is constructed leveraging a library of joint types that define the holonomic bilateral constraints. As a results, systems that satisfy the limitations of a given recursive method typically enjoy a more automated modeling process that is generally less error prone, faster to iterate with, and faster computationally to solve.

Again, the new optimal design frameworks developed in this research effort are independent of which dynamics formulation is selected, therefore, it is functional with this new class of *O(n)* and *O(log(n))* methods.

2.7 Kinematic Constraints

Kinematically assembled systems are constrained by nonlinear equality constraints (or bilateral constraints) of the form,

$$\mathbf{s} = \mathbf{\Phi}^{\mathrm{ol}}(\boldsymbol{\rho}, \boldsymbol{\theta}) \tag{1}$$

for open-loop relations; where $\boldsymbol{\theta} \in \mathbb{R}^{n_p}$ are independent system parameters, $\boldsymbol{\rho} \in \mathbb{R}^{n_{dv}}$ are dependent system parameters, and $\boldsymbol{s} \in \mathbb{R}^{n_o}$ represents kinematic assembly relations such as gaps, positions, and orientations of subcomponent features [155]. (However, in this specific case, $\boldsymbol{\theta}$ and $\boldsymbol{\rho}$ are independent and dependent kinematic assembly variables, respectively.) The dependent assembly variables, $\boldsymbol{\rho}$, must satisfy all closed kinematic constraints; these may be defined as,

$$\mathbf{\Phi}^{\mathrm{cl}}(\boldsymbol{\rho}, \boldsymbol{\theta}) = \mathbf{0} \tag{2}$$

Equations (1)–(2) are implicitly accounted for in the ordinary differential equations (ODEs), however, they are explicitly included in constrained dynamical systems described by differential algebraic equations (DAEs)—as discussed in the following sections.

2.8 Ordinary Differential Equations (ODEs)

The primary contribution of this research effort is in the formulation of the new optimal design frameworks for uncertain dynamical systems, not in the formulations for dynamical equations of motion (EOMs). Therefore, case studies described by ODEs will be presented using the traditional Euler/Lagrange multibody dynamics formulation [115-117, 119, 147]. As a very brief overview, the Euler-Lagrange ODE formulation for a multibody dynamic system can be described by,

$$\begin{gathered}\boldsymbol{M}\big(\boldsymbol{q}(t), \boldsymbol{\theta}(t)\big)\dot{\boldsymbol{v}}(t) + \boldsymbol{C}\big(\boldsymbol{q}(t), \boldsymbol{v}(t), \boldsymbol{\theta}(t)\big)\boldsymbol{v}(t) + \boldsymbol{N}\big(\boldsymbol{q}(t), \boldsymbol{v}(t), \boldsymbol{\theta}(t)\big) \\ = \boldsymbol{\mathcal{F}}\big(\boldsymbol{q}(t), \boldsymbol{v}(t), \dot{\boldsymbol{v}}(t), \boldsymbol{\theta}(t)\big) = \boldsymbol{\tau}(t)\end{gathered} \tag{3}$$

where $\boldsymbol{q}(t) \in \mathbb{R}^{n_d}$ are independent generalized coordinates equal in number to the number of degrees of freedom, n_d; $\boldsymbol{v}(t) \in \mathbb{R}^{n_d}$ the generalized velocities and—using Newton's *dot* notation—$\dot{\boldsymbol{v}}(t)$ contains their time derivatives; $\boldsymbol{\theta}(t) \in \mathbb{R}^{n_p}$ includes independent system parameters which may or may not be functions of time; $M\big(\boldsymbol{q}(t), \boldsymbol{\theta}(t)\big) \in \mathbb{R}^{n_d \times n_d}$ is the inertia matrix; $\boldsymbol{C}\big(\boldsymbol{q}(t), \boldsymbol{v}(t), \boldsymbol{\theta}(t)\big) \in \mathbb{R}^{n_d \times n_d}$ includes centrifugal, gyroscopic and Coriolis effects; $\boldsymbol{N}\big(\boldsymbol{q}(t), \boldsymbol{v}(t), \boldsymbol{\theta}(t)\big) \in \mathbb{R}^{n_d}$ the generalized gravitational and joint forces; and $\boldsymbol{\tau}(t) \in \mathbb{R}^{n_i}$ are the n_i applied wrenches.

> **Remark 1**
> For notational brevity the explicit dependence on time will be dropped unless an occasion calls for stating it explicitly.

The relationship between the time derivatives of the independent generalized coordinates and the generalized velocities is,

$$\dot{\boldsymbol{q}} = \boldsymbol{H}(\boldsymbol{q}, \boldsymbol{\theta})\boldsymbol{v} \tag{4}$$

where $\boldsymbol{H}(\boldsymbol{q}, \boldsymbol{\theta})$ is a skew-symmetric matrix that is a function of the selected kinematic representation (e.g. Euler Angles, Tait-Bryan angles, Axis-Angle, Euler Parameters, etc.) [120, 156]. One of the contributions of this work is an optimal design framework that accounts for the more general systems where $\dot{\boldsymbol{q}} \neq \boldsymbol{v}$. None of the literature reviewed for the optimal design of multibody dynamic systems during this work treats the more general case enabled by (4) as it complicates the formulation of the optimal design framework.

The trajectory of the system is determined by solving (3)–(4) as an initial value problem, where $\boldsymbol{q}(\boldsymbol{0}) = \boldsymbol{q}_0$ and $\boldsymbol{v}(\boldsymbol{0}) = \boldsymbol{v}_0$. Also, the system measured outputs are defined by,

$$\boldsymbol{y} = \boldsymbol{\mathcal{O}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\theta}) \tag{5}$$

where $\boldsymbol{y} \in \mathbb{R}^{n_o}$ with n_o equal to the number of outputs. Outputs can represent the generalized coordinates or their rates, the generalized velocities, or any function of these quantities (e.g. generalized accelerations).

2.9 Differential Algebraic Equations (DAEs)

This section briefly introduces the DAE formulation presented by Haug in [120], where constrained multibody dynamics can be expressed by,

$$\begin{bmatrix} \boldsymbol{M}(\boldsymbol{q},\boldsymbol{\theta}) & \boldsymbol{\Phi}_{\mathbf{q}}^{\mathbf{T}} \\ \boldsymbol{\Phi}_{\mathbf{q}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{v}} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\tau} \\ \boldsymbol{\alpha} \end{bmatrix} \tag{6}$$

with $(\cdot)_*$ representing a jacobian with respect to *; $\boldsymbol{\Phi}(\boldsymbol{q},\boldsymbol{\theta},t) = \{(\boldsymbol{\Phi}^{\mathrm{ol}} - \mathbf{s}), \boldsymbol{\Phi}^{\mathrm{cl}}\} = \mathbf{0}$ are bilateral constraints, or joints; $\boldsymbol{\lambda}$ are lagrange multipliers; and $\boldsymbol{\alpha}$ are the associated acceleration constraints equal to $\boldsymbol{\alpha} = -(\boldsymbol{\Phi}_{\mathbf{q}}\dot{\boldsymbol{q}})_{\mathbf{q}} - \mathbf{2}\boldsymbol{\Phi}_{\mathbf{q}\mathrm{t}} - \boldsymbol{\Phi}_{\mathrm{tt}} = \boldsymbol{\Phi}_{\mathbf{q}}\ddot{\boldsymbol{q}}$; and $\boldsymbol{q}(t) \in \mathbb{R}^{n_{gc}}$ are the generalized coordinates with $n_{gc} \geq n_d$. In a DAE formulation, the generalized coordinates, $\boldsymbol{q}$, are no longer independent and therefore constraint forces that couple the dependent coordinates must be calculated. These constraint forces are equal to the Lagrange multipliers, $\boldsymbol{\lambda}$.

Combining (4)–(6) yields an *index–1* initial value DAE problem for constrained multibody dynamic systems. The numerical solution of (6) is significantly more challenging than the ODEs defined in (3) and the interested reader is referred to [157] for more information.

As discussed in Section 8.2, the optimal design of uncertain DAEs is out of the scope of this research and is suggested as a future research direction.

2.10 Autonomous and Nonautonomous Systems

Differential equations that are not explicit functions of time are referred to as *autonomous*, or *time-invariant* systems. However, with an explicit dependence on time they are referred to as *nonautonomous*, or *time-varying* systems. *Nonautonomous* systems can be converted to an *autonomous* system by appending a new time variable to the list of state variables, $\{q_1, \ldots, q_{n_s}, t\}$, and appending the relation $\dot{t} = 1$ to the list of differential equations.

3 Numerical Optimization

3.1 Motivation

This chapter presents various classifications of numerical optimization techniques followed by a very brief discussion of some practical implementation considerations. The purpose in reviewing this material is to clearly define the type of numerical optimization the new computational frameworks were developed for. The new frameworks are scoped such that only smooth constrained dynamical systems are addressed; thus targeting local constrained nonlinear optimization techniques as the primary solver to be used. Global optimization and Directed Search (DS) techniques may be appropriately applied to the problems addressed by this work; however, this work does not include any results from these methods.

The interested reader is referred to [158-161] for a more comprehensive introduction to the topics presented in this chapter.

3.2 Solver Families

Numerical optimization solvers can be classified into four primary groups: *Linear*, *Nonlinear*, *Global*, and *Direct Search* solvers. Since the new optimal design frameworks address general nonlinear dynamical systems, linear optimization techniques will not be reviewed. The brief discussion of these solver classes will proceed in the reversed order.

3.2.1 Direct Search

The *Direct Search* techniques have some important qualities due to their heuristic and non-gradient-based sampling approaches: first, they are *likely* to avoid getting caught in local minimum; second, they can handle discontinuous cost functions and constraints; third, they are less efficient due to the increased sampling induced by not using gradient information; fourth, they cannot provide a guarantee of finding a *global minimum*.

Many methods from this class include an element of random sampling and are frequently referred to as *stochastic solvers*. Others emulate behaviors or attributes found in nature and therefore are referred to as *meta-heuristic* methods.

The literature presents a large variety of *Direct Search* methods, however, only a modest sampling of some representative methods are presented.

3.2.1.1 Nelder-Mead

The Nelder-Mead (NM) method is quite simple in that it generates an n-dimensional polytope, or simplex (not to be confused with the Simplex algorithm used in Linear Programming), with n+1 points forming its vertices [162, 163]. At each step the functional values of these vertex points are sorted low to high. The worst, or highest, point is then reflected through the centroid of the polytope. Various additional heuristics are used in determining the length of the reflected point. As long as the new point is "better" than its previous value, then it's kept as a new point in the simplex. The process proceeds until the simplex shrinks to within a predefined volume. This method is a deterministic method.

3.2.1.2 Dividing Rectangles

The Divided Rectangles (DIRECT) method [164] is a deterministic method that systematically divides the search space into successively smaller hyper-rectangles until a convergence criterion is satisfied. There is a *local* formulation that provides faster convergence but is more likely to stop in a local minimum [165].

3.2.1.3 Differential Evolution

Differential Evolution (DE) is a relatively new Evolutionary Algorithm and was created by Storn and Price in 1996 [166]. It has a similar concept to that of Genetic Algorithms but the stochastic evolution process is new. If the search includes n design variables, then a population of m sample points is randomly generated; where $m \gg n$. There are two main phases of a given DE step. First, three points are randomly selected from the population, $\{x_a, x_b, x_c\}$, to aid in creating a temporary *evolution point*, x_e. The *evolution point* is created from the following relationship,

$$x_e = x_a + s(x_b - x_c)$$

where s is a scaling factor that ensures shrinkage of the population over time. The next phase includes selecting another random point from the population, x_p, which will compete with a newly generated point, x_n, to 'stay alive'. The new point, x_n, is created by probabilistically selecting each of its coordinates from either x_p or x_e. If $f(x_n) < f(x_p)$ then x_n joins the population and x_p is discarded. This process proceeds until the distance between the new best point and old best point drops below some prescribed tolerance.

3.2.1.4 Simulated Annealing

Simulated Annealing (SA) was developed in the early 1980's and was inspired by the annealing process found in metallurgy [163, 167]. It is implemented with multiple starting points where each point goes through the following process. At each step of the solver a new point, x_n, is randomly generated in the neighborhood of the current point x_i. The radius of the neighborhood around x_i, where x_n is created, shrinks at a specified rate each step of the solver. The *best* point, x_b, visited during the history of steps associated with x is maintained. Each newly generated point is compared with the current value of x_i and x_b. If $f(x_n) < f(x_b)$ then x_n is kept as the new x_b. However, if $f(x_b) < f(x_n) < f(x_i)$, then x_n probabilistically replaces x_i based on some predefined relation.

3.2.1.5 Random Search

The Random Search (RS) method is very straight forward; a sample of m start points is randomly distributed in the search space. Each start point then proceeds using a selected local search method, such as Interior Point (IP), Active Set (AS), or Sequential Quadratic Programming (SQP). This technique simply tries to increase the likelihood of finding the global by having more starting points, however, its success is completely determined by the random placement of its start points and can be very costly computationally.

Random Search, Differential Evolution, and Simulated Annealing lend themselves to a parallel implementation where many test points can be generated and evolved over time in parallel, thus allowing for a more thorough and efficient coverage of the search space.

3.2.2 Global Optimization

The distinction between the nonlinear and global methods is: nonlinear methods do not directly attempt to find the *global minimum*. Instead, they stop at the first minimum they find. If a system is nonconvex, a nonlinear solver gets “stuck” in a *local minimum*. For this reason, nonlinear solvers are frequently referred to as *Local Searches*. In contrast, a global method makes an attempt to find the true *global minimum*.

The topic of Global Optimization is a quite deep and an active area of research. Referring to global methods as a complete search, Dr. Arnold Neumaier describes it best,

> *“In contrast to local or heuristic searches, a complete search checks all points in the search region for feasibility, and all feasible points for global optimality. A solver that performs a complete search—apart from rounding error issues—is called a complete solver. Since the search region for continuous global optimization problems contains an infinite number of points, analytic techniques are needed to make decisions about infinitely many points simultaneously. This is usually (but not always) done in a branch-and-bound framework.”* [168]

Various techniques exist where *interval methods* and *convex analysis* play a major roll. The interested reader is referred to [158, 168-171] for a more comprehensive introduction to the topic.

3.2.3 Nonlinear Programming

Nonlinear optimization is frequently referred to as Nonlinear Programming (NLP) and local searches. Not all NLP methods make use of gradient information, however, all have assumptions of convexity to some level and are therefore limited to a local search when these assumptions are not met.

Many NLP methods are available in the literature, however, only a modest sampling of some representative methods will be reviewed in this work.

3.2.3.1 Interior Point

The Interior Point (IP) algorithm converts a objective function with inequalities to a new objective function with equality constraints using slack variables and barrier methods [172]. The dual problem is then constructed and solved by linearizing the nonlinear terms. During each iteration a line search is conducted to determine the step length along a direction created using the Hessian of the *Lagrangian*. The search continues until the difference in search variables from one time step to the next is less than the given tolerance.

3.2.3.2 Sequential Quadratic Programming

The Sequential Quadratic Programming (SQP) technique is a commonly used method in constrained NLP [160]. It can be summarized as repeatedly creating second order approximates of the cost function and linear approximations of the constraints at each step of the solver. The approximate system is then solved with traditional Quadratic Programming (QP) techniques. Specifically, improved speed is accomplished by using the quasi-Newton BFGS technique to approximation of the Hessian, and use of a *merit function* in parallel with the QP solvers guides each iteration and improves convergence.

3.3 Sequential vs. Simultaneous Formulations

NLPs involving differential equations (DEs) in the equality constraints may be formulated in one of two predominate methodologies, namely: sequentially, or simultaneously [65, 66]. (The literatures occasionally refers to the sequential approach as *partial discretization* and to the simultaneous as *full discretization* [173].)

In the *sequential NLP* (SeqNLP) approach, the dynamical equations (3)–(5) remain as continuous functions that may be integrated with standard off-the-shelf ODE solvers (such as Runge-Kutta). The integrated states are then used to evaluate the system outputs, constraints and objective function. The SeqNLP methodology results in a smaller optimization problem as only the optimization search/manipulated variables $\boldsymbol{x}$ are discretized.

A *simultaneous NLP* (SimNLP), however, discretizes the DEs over the time horizon as explicit difference equations that become equality constraints for the NLP. As a result, the discretized states are added to the optimizer's list of the manipulated variables. For example, assume a 1-DOF 2nd-order mass-spring-damper system is being optimized by manipulating its spring and damping coefficients. Also, assume the allowable coefficient values are bounded. This results in two 1st-order ODEs as equality constraints; four inequality constraints, and two manipulated variables for the NLP. Further, assume the time horizon of the problem is from [0,1] seconds; the required time step is 1/10 of a second; and the system is discretized with an Euler method. This would result in 20 difference equations and 20 new state variables; bringing the total number of manipulated variables up to 22, inequality constraints to 24, and equality constraints to 20. As can be seen, the SimNLP can rapidly expand the size of the problem through the discretization process. However, added structure is gained and SimNLP's typically converge faster and are more robustly solved.

3.4 Hard vs. Soft Constraints

An optimization practitioner has a few options when prescribing constraints. *Hard* constraints are specified in the list of *such that* conditions of a given NLP formulation. These constraints typically take the solver longer to satisfy. In contrast, *soft* constraints are specified in the cost function as a heavily weighted term. These constraints are typically easier to satisfy but may only be loosely satisfied.

Soft constraints may be implemented in a number of ways [160]. Suggested techniques for both inequality and equality constraints are now presented.

3.4.1 Soft Inequality Constraints

Soft inequality constraints can be accomplish through a penalty term of the form,

$$\varsigma \max\left(0, h(\mathrm{x}, bound)\right)^2 \tag{7}$$

where ς is a large constant and h is some function of the variable x that is being bounded. With a large ς, this relationship is analogous to an inequality like penalizing term, meaning, if the variable *x* is outside its bounds—or outside the feasible region—then it's heavily penalized. When it's within the feasible region there is no penalty. Also, by squaring the *max* function, its discontinuity is smoothed out (a desirable feature for gradient based solvers).

3.4.2 Soft Equality Constraints

When establishing a *soft* equality relation, such as $a = b$, then a penalty term of the following form may be used,

$$\varsigma\ (\mathrm{a}-\mathrm{b})^2 \tag{8}$$

Thus, by providing a large ς any difference between variables *a* and *b* is heavily penalized and driven down by the solver.

3.5 Multi-Objective Optimization (MOO)

When multiple terms exist in the objective function, $\mathrm{J_{obj}}$, the optimum solution exists on a curve, or surface, known as the *Pareto Optimal Sets*. The surface is a function of the relative weights, $\{\mathrm{w_1}, \mathrm{w_2}, \dots, \mathrm{w_n}\}$, given to each respective term in the objective function.

$$\mathrm{J_{obj}} = \mathrm{w_1 J_1} + \mathrm{w_2 J_2} + \cdots + \mathrm{w_n J_n} \tag{9}$$

Therefore, any point on the *Pareto Optimal Sets* is optimal. Which point on the surface is selected by the solver depends on the relative weights for each respective term in the objective function. In the end, the practitioner must provide the insight to properly select the relative weights. Problems of this nature are frequently referred to as *multi-objective optimization* (MOO) problems.

3.5.1 Constrained Multi-Objective Optimization (cMOO)

A Pareto optimal set may also be found as active constraint boundaries are moved [89, 174]. For example, if a given output, y_1**,** is bounded from above by, $\overline{y}$, we have the following constraint, $y_1 \leq \overline{y}$. A Pareto set will be obtained from the optimal design for unique values of $\overline{y}$ as long as the constraint is active. Once the constraint becomes inactive the constraint has no influence on the optimal value. Occasionally, a MOO problem may be appropriately rewritten such that extra objective terms, say $\{\mathrm{J_2}, \dots, \mathrm{J_n}\}$, are redefined as *hard* constraints. For example,

$$\begin{array}{cc} \underline{\mathrm{J}}_2 - \mathrm{J}_2 \leq 0, & \mathrm{J}_2 - \overline{\mathrm{J}}_2 \leq 0 \\ & \vdots \\ \underline{\mathrm{J}}_\mathrm{n} - \mathrm{J}_\mathrm{n} \leq 0, & \mathrm{J}_\mathrm{n} - \overline{\mathrm{J}}_\mathrm{n} \leq 0 \end{array} \tag{10}$$

where $\left\{\underline{\mathrm{J}}_i, \overline{\mathrm{J}}_i\right\}$ represents lower and upper bounds. In doing so, the Pareto set is now governed by the unique values of the bounds of the active constraints. This reformulation of MOO will be referred to as a *constrained multi-objective optimization* (cMOO) problem.

4 Modeling Uncertainty in Algebraic and Differential Systems

4.1 Motivation

This chapter reviews the Generalized Polynomial Chaos (gPC) technique that is used to quantify uncertainty in the new optimal design frameworks presented in subsequent chapters. Generalized Polynomial Chaos offers a way to build uncertain differential equations (UDEs) that are very efficient, relative to contemporary methods, and thus enable a computationally feasible optimal design framework for uncertain dynamical systems. The presentation that follows is heavily influenced by [5, 7, 9, 10, 25].

4.2 Generalized Polynomial Chaos

Generalized Polynomial Chaos (gPC), first introduced by Wiener [4], is an efficient method for analyzing the effects of uncertainties in second order random processes [5]. This is accomplished by approximating an independent source of uncertainty, θ, with an infinite series of weighted orthogonal polynomial bases called Polynomial Chaoses. Clearly, an infinite series is impractical; therefore, a truncated set of $p_o + 1$ terms is used with $p_o \in \mathbb{N}$ representing the *order* of the approximation. Or,

$$\theta(\xi) = \sum_{j=0}^{p_o} \theta^j \psi^j\,(\xi) \tag{11}$$

where $\theta^j \in \mathbb{R}$ are known coefficients; and $\psi^j \in \mathbb{R}$ are individual single dimensional orthogonal basis terms (or modes). The bases are orthogonal with respect to the ensemble average inner product,

$$\langle \psi^i(\xi), \psi^j(\xi) \rangle \; = \; \int_{\Omega} \psi^i(\xi)\psi^j(\xi)f(\xi)d\xi \; = \; 0, \quad \text{for i≠j} \tag{12}$$

where $\xi(\omega) \in \mathbb{R}$ is a random variable that maps the random event $\omega \in \Omega$, from the sample space, Ω, to the domain of the orthogonal polynomial basis (e.g., $\xi\colon \Omega \to [-1,1]$); $f(\xi)$ is the weighting function that is equal to the joint probability density function of the random variable.

Also, $\langle \psi^j, \psi^j \rangle = 1, \forall j$ when using *normalized basis*; *standardized basis* are constant and may be computed off-line for efficiency using (12).

Remark 2
For notational brevity the explicit dependence between the random variable, ξ, and the random event, ω, will not be shown throughout the remainder of this work.

Table 4.1—Selected distribution/orthogonal polynomial pairs

Uncertainty Distribution	Orthogonal Polynomial	Domain
Gaussian	Hermite	$[-\infty, \infty]$
Beta	Jacobi	$[-1,1]$
Uniform	Legendre	$[-1,1]$
Gamma	Laguerre	$[0, \infty]$

The choice of basis to be used is dependent on the type of statistical distribution that best models a given uncertain parameter. In [5], a family of orthogonal polynomials and statistical distribution pairs was presented. (Table 4.1 presents a selected subset of example paired distributions and orthogonal polynomials.) Therefore, gPC allows a designer to pick an appropriate distribution and polynomial pair to model the uncertainty. For example, the Normal, or Gaussian, distribution, denoted by $\mathcal{N}(\mu, \sigma)$, and the Uniform distribution, denoted by $\mathcal{U}(a, b)$, are commonly used distributions; where μ is the mean, σ is the standard deviation, and a and b are the lower and upper bounds of the distribution range, respectively. When modeling an uncertainty with $\mathcal{N}(\mu, \sigma)$ the corresponding expansion basis is the Hermite polynomials, $H_n(\xi)$, and the expansion with known coefficients is,

$$\theta(\xi) = \mu\, H_0(\xi) + \ \sigma\, H_1(\xi) + 0\, H_2(\xi) + \cdots + \ 0\, H_{p_o}(\xi) \tag{13}$$

where the domain is $\xi\colon \Omega \to [-\infty, \infty]$. Similarly, when the uncertainty is better modeled with $\mathcal{U}(a, b)$, then Legendre polynomials are used, $L_n(\xi)$, and the expansion with known coefficients is,

$$\theta(\xi) = \left(a + \frac{b-a}{2}\right) L_0(\xi) + \left(\frac{b-a}{2}\right) L_1(\xi) + 0\, L_2(\xi) + \cdots + \ 0\, L_{p_o}(\xi) \tag{14}$$

where the domain is $\xi\colon \Omega \to [-1,1]$. (For more information regarding possible distribution/polynomial pairs the interested reader should refer to [5].)

Any quantity dependent on a source of uncertainty becomes uncertain and can be approximated in a similar fashion as (11),

$$\rho(\theta(\xi)) = \rho(\xi) = \sum_{j=0}^{n_b} \rho^{\mathrm{j}}\, \Psi^j\,(\xi) \tag{15}$$

where $\rho(\xi)$ is an approximated dependent quantity; ρ^{j} are the unknown gPC expansion coefficients; and $n_b \in \mathbb{N}$ is the number of basis terms in the approximation.

The orthogonal basis may be multidimensional in the event that there are multiple sources of uncertainty. The multidimensional basis functions are represented by $\Psi^{\mathrm{j}} \in \mathbb{R}^{n_b}$. Additionally, $\boldsymbol{\xi}$ becomes a vector of random variables, $\boldsymbol{\xi} = \{\xi_1, \dots, \xi_{n_p}\} \in \mathbb{R}^{n_p}$, and maps the sample space, Ω, to an n_p dimensional cuboid, $\boldsymbol{\xi}: \Omega \rightarrow [-1,1]^{n_p}$ (as in the example of Legendre chaoses).

The multidimensional basis is constructed from a product of the single dimensional basis in the following manner,

$$\Psi^j = \psi_1^{i_1} \psi_2^{i_2} \dots \psi_{n_p}^{i_{n_p}}, \quad i_k = 0 \dots p_o, k = 1 \dots n_p \tag{16}$$

where subscripts represent the uncertainty source and superscripts represent the associated basis term (or mode). A complete set of basis may be determined from a full tensor product of the single dimensional bases. This results in an excessive set of $(p_o + 1)^{n_p}$ basis terms. Fortunately, the multidimensional sample space can be spanned with a minimal set of $n_b = (n_p + p_o)!/(n_p!\ p_o!)$ basis terms. The minimal basis set can be determined by the products resulting from these index ranges,

$$\begin{gathered} i_1 = 0 \dots p_o, \\ i_2 = 0 \dots (p_o - i_1), \\ \vdots \\ i_{n_p} = 0 \dots \left(p_o - i_1 - i_2 - \dots - i_{(n_p - 1)}\right) \end{gathered}$$

The number of multidimensional terms, n_b, grows quickly with the number of uncertain parameters, n_p, and polynomial order, p_o. Sandu et. al. showed that gPC is most appropriate for modeling systems with a relatively low number of uncertainties [9, 17] but can handle large nonlinear uncertainty magnitudes.

Once all sources of uncertainty, $\theta(\xi)$, and dependent quantities, $\rho(\xi)$, have been expanded then the constitutive relations defining a given problem may be updated. For example, constitutive relations may be algebraic or differential equations,

$$\mathcal{A}(\rho(\xi), \theta(\xi)) = 0 \tag{17}$$

$$\dot{\rho}(\mathrm{t}; \xi) = \mathfrak{D}(\rho(\mathrm{t}; \xi), \theta(\xi)) \tag{18}$$

Equation (17) is an implicit algebraic constitutive relation and (18) is a differential constitutive relation. It is important to note that the dependent quantities are functions of time in (18), therefore, (15) is modified to,

$$\rho(t, \theta(\xi)) = \rho(t; \xi) = \sum_{j=0}^{n_b} \rho^{\mathrm{j}}(t)\, \Psi^j\,(\xi) \tag{19}$$

It is instructive to notice how time and randomness are decoupled within a single term of the gPC expansion. Only the expansion coefficients are dependent on time, and only the basis terms are dependent on the n_b random variables, $\boldsymbol{\xi}$. If any sources of uncertainty are also functions of time then (11) needs to be updated in a similar fashion as (19) and then all dependent quantities will have to be expanded using (19). Without loss of generality, the proceeding presentation will assume that all sources of uncertainty are time-independent to simplify the notation.

Substituting the appropriate expansions from (11), (15), or (19) into the constitutive relations results in uncertain constitutive equations,

$$\mathcal{A}\left(\sum_{j=0}^{n_b} \rho^{\mathrm{j}}(t)\, \Psi^j\,(\xi), \sum_{j=0}^{p_o} \theta^j \psi^j\,(\xi)\right) = 0 \tag{20}$$

$$\sum_{j=0}^{n_b} \dot{\rho}^{\mathrm{j}}(t; \xi)\, \Psi^j\,(\xi) = \mathfrak{D}\left(\sum_{j=0}^{n_b} \rho^{\mathrm{j}}(t)\, \Psi^j\,(\xi), \sum_{j=0}^{p_o} \theta^j \psi^j\,(\xi)\right) \tag{21}$$

where the n_b expansion coefficients, ρ^{j} or $\rho^{\mathrm{j}}(t)$, from the dependent quantities are the unknowns to be solved for.

There are a number of methods in the literature for solving equations such as (20)–(21). The Galerkin Projection Method (GPM) takes advantage of the orthogonal relations of the basis polynomials by projecting (20)–(21) onto the *n*th basis through the associated ensemble inner product (12).

$$\langle \psi^n(\xi), \sum_{j=0}^{n_b} \dot{x}_i^j(t)\Psi^j(\boldsymbol{\xi})\rangle = \langle \psi^n(\xi), f\left(\sum_{j=0}^{n_b} x_i^j(t)\Psi^j(\boldsymbol{\xi}), \sum_{j=0}^{p_o} \theta_k^j(t)\, \psi_k^j(\xi_k)\right)\rangle \tag{22}$$

This results in a single expanded deterministic system of equations that can be solved using traditional methods. The GPM enjoys an exponential convergence to the exact solution as $p_o \to \infty$. However, the GPM is very intrusive and difficult to implement for an arbitrarily complex system. As an alternative, sample-based collocation techniques can be used without the need to modify the base equations.

Sandu et. al. [9, 10] showed that the collocation method solves equations such as (20)–(21) by solving (17)–(18) at a set of points, ${}_k\boldsymbol{\mu} \in \mathbb{R}^{n_p}$, $k = 1 \dots n_{cp}$, selected from the n_p dimensional domain of the random variables $\boldsymbol{\xi} \in \mathbb{R}^{n_p}$. Meaning, at any given instance in time, the random variables' domain is sampled and solved n_{cp} times with $\boldsymbol{\xi} = {}_k\boldsymbol{\mu}$ (updating the approximations of all sources of uncertainty for each solve), then the uncertain coefficients of the dependent quantities can be determined. This can be accomplished by defining intermediate variables such as,

$$\mathcal{P}({}_k\boldsymbol{\mu}) = \sum_{j=0}^{n_b} \rho^{\mathrm{j}}\Psi^j(\xi), \;\; k = 0 \dots n_{cp} \tag{23}$$

Substituting the appropriate intermediate variables into (20) and (21) respectively yields,

$$\mathcal{A}\left(\mathcal{P}({}_k\boldsymbol{\mu}),\, {}_k\Theta_r({}_k\boldsymbol{\mu})\right) = 0 \tag{24}$$

$$\dot{\mathcal{P}}({}_k\boldsymbol{\mu}) = \mathfrak{D}\left(\mathcal{P}({}_k\boldsymbol{\mu}),\, {}_k\Theta_r({}_k\boldsymbol{\mu})\right) \tag{25}$$

where $k = 0 \dots n_{cp}, r = 1 \dots n_p$, and each uncertainty's intermediate variable is,

$${}_k\Theta_r({}_k\boldsymbol{\mu}) = \sum_{j=0}^{p_o} \theta_r^j\, \psi_k^j({}_k\boldsymbol{\mu}) \tag{26}$$

Remark 3:

Equation (26) represents updating the approximation for each source of uncertainty for the given collocation point $_{k}\boldsymbol{\mu}$. In (26) we show the individual sources of uncertainty are expanded with their respective single dimensional basis, ψ_k^j. Using this notation requires the corresponding element—say the *i*th, from the *k*th collocation point, $_{k}\boldsymbol{\mu}$—to be indexed for the expansion of the given uncertainty. An alternative method is to expand the uncertainty with the full multidimensional basis, Ψ^j, and apply the full collocation point $_{k}\boldsymbol{\mu}$ in the update. Use of this second method requires additional book-keeping during implementation to properly map the known $\theta_k^j(t)$ for the given uncertainty to their respective single dimensional basis ψ_k^j terms in Ψ^j. However, either method is valid.

Equations (24)–(25) provide a set of n_{cp} independent equations whose solutions determine the uncertain expansion coefficients of the dependent quantities. Since (24) is implicitly defined, there are two options in determining $\mathcal{P}(\,_{k}\boldsymbol{\mu})$: use a numerical nonlinear system solver such as Newton-Raphson, or, solve (17) for a new relation that defines $\mathcal{P}(\,_{k}\boldsymbol{\mu})$ explicitly. The uncertain expansion coefficients of the dependent quantities are determined by recalling the relationship of the expansion coefficients to the solutions as in (23). In matrix notation, (23) can be expressed as,

$$_{\mathrm{k}}\mathcal{P} = \boldsymbol{\Psi}(\,_{k}\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\rho} \tag{27}$$

where the matrix,

$$A_{k,j} = \Psi^j(\,_{k}\mu),\ \ j = 0 \dots n_b, k = 0 \dots n_{cp} \tag{28}$$

is defined as the *collocation matrix*. It's important to note that $n_b \leq n_{cp}$. The expansion coefficients can now be solved for using (27),

$$\boldsymbol{\rho} = \boldsymbol{A}^{\#}\boldsymbol{\mathcal{P}} \tag{29}$$

where $A^{\#}$ is the pseudo inverse of A if $n_b < n_{cp}$. If $n_b = n_{cp}$, then (29) is simply a linear solve. However, [10, 175-178] presented the least-squares collocation method (LSCM) where the stochastic dependent variable coefficients are solved for, in a least squares sense, using (29) when $n_b < n_{cp}$. Reference [10] also showed that as $n_{cp} \to \infty$ the LSCM approaches the GPM solution; by selecting $3n_b \leq n_{cp} \leq 4n_b$ the greatest convergence benefit is achieved with minimal computational cost. LSCM also enjoys the same exponential convergence rate as $p_o \to \infty$.

The nonintrusive nature of the LSCM sampling approach is arguably its greatest benefit; (17) or (18) may be repeatedly solved without modification. Also, there are a number of methods for selecting the collocation points and the interested reader is recommended to consult [6-10] for more information.

Once the expansion coefficients of the dependent quantities are determined then statistical moments, such as the expected value and variance, can be efficiently calculated. Arguably the greatest benefit of modeling uncertainties with gPC is the computational efficiencies gained when calculating the various statistical moments of the dependent quantities. For example, [1] defines the statistical expected value as,

$$\mu_\rho = E[\rho(\boldsymbol{\xi})] = \int_\Omega \rho(\boldsymbol{\xi})\, f(\boldsymbol{\xi}) d\boldsymbol{\xi} \tag{30}$$

and the variance,

$$\sigma_\rho^2 = Var[\rho(\boldsymbol{\xi})] = \int_\Omega \left(\rho(\boldsymbol{\xi}) - \mu_\rho\right)^2 f(\boldsymbol{\xi}) d\boldsymbol{\xi} = E\left[\left(\rho(\boldsymbol{\xi}) - \mu_\rho\right)^2\right] \tag{31}$$

with the standard deviation, $\sigma_\rho = \sqrt{Var[\rho(\boldsymbol{\xi})]}$. Given these definitions and leveraging the orthogonality of the gPC basis, these moments may be efficiently computed by a reduced set of arithmetic operations of the expansion coefficients,

$$\mu_\rho = E[\rho(\boldsymbol{\xi})] = \rho^0 \langle \Psi^0, \Psi^0 \rangle \tag{32}$$

$$\sigma_\rho = \sqrt{E\left[\left(\rho(\boldsymbol{\xi}) - \mu_\rho\right)^2\right]} = \sqrt{\sum_{j=1}^{n_b} (\rho^j)^2 \langle \Psi^j, \Psi^j \rangle} \tag{33}$$

Also, recall that $\langle \Psi^j, \Psi^j \rangle = 1, \forall j$ when using *normalized basis*; *standardized basis* are constant and may be computed off-line for efficiency using (12). A number of efficient statistical moments may be determined from the expansion coefficients. The authors presented a number of gPC based measures using efficient moments such as (32)–(33) in [78-81, 104].

To summarize, the following basic steps are taken in order to model uncertainty and solve for statistical moments of quantities dependent on uncertainties:

1. Model all sources of uncertainty by associating an appropriate probability density function (PDF).
2. Expand all sources of uncertainty with an appropriate single dimensional orthogonal polynomial basis. The known expansion coefficients are determined from the PDF modeling the uncertainty.
3. Expand all dependent quantities with an appropriately constructed multi-dimensional basis.
4. Update constitutive relations with the expansions from steps 2–3. The new unknowns are now the expansion coefficients from the dependent quantities.
5. Solve the uncertain constitutive relations for the unknown expansion coefficients; this work uses the LSCM technique.
6. Calculate appropriate statistical moments from the expansion coefficients.

The following sections summarize this material in the context of: uncertain kinematic assemblies, and uncertain dynamical systems.

4.2.1 Uncertain Kinematic Assemblies

To apply gPC to algebraic kinematic assemblies, as described by (1)–(2), the independent sources of uncertainty, $\boldsymbol{\theta}(\boldsymbol{\xi})$, are either geometric dimensions or surface finishes. The dependent uncertain quantities, $\boldsymbol{\rho}(\boldsymbol{\xi})$, are the dependent dimensions in (1)–(2).

Therefore, all independent and dependent quantities and assembly features need to be approximated by (11), (15) or (19), as appropriate. Substituting these approximations into the algebraic constraints (1)–(2) yields,

$$\sum_{j=0}^{n_b} \mathbf{s}^j \Psi^j = \boldsymbol{\Phi}^{\mathrm{ol}} \left(\sum_{j=0}^{n_b} \boldsymbol{\rho}^j \Psi^j , \sum_{j=0}^{p_o} \theta_k^j \, \psi_k^j \right) \tag{34}$$

$$\boldsymbol{\Phi}^{\mathrm{cl}} \left(\sum_{j=0}^{n_b} \boldsymbol{\rho}^j \Psi^j , \sum_{j=0}^{p_o} \theta_k^j \, \psi_k^j \right) = \mathbf{0} \tag{35}$$

where, $k = 1 \dots n_p$. Applying the preferred LSCM to (34)–(35) requires defining intermediate variables for the dependent quantities as in (23). Substituting these intermediate variables along with (26) into (34)–(35) yields,

$$ {}_k\boldsymbol{S}(t;\ {}_k\boldsymbol{\mu}) = \boldsymbol{\Phi}^{\mathrm{ol}}\left({}_k\boldsymbol{\mathcal{P}}(t;\ {}_k\boldsymbol{\mu}),\ {}_k\boldsymbol{\Theta}(t;\ {}_k\boldsymbol{\mu})\right) \tag{36} $$

$$ \boldsymbol{\Phi}^{\mathrm{cl}}\left({}_k\boldsymbol{\mathcal{P}}(t;\ {}_k\boldsymbol{\mu}),\ {}_k\boldsymbol{\Theta}(t;\ {}_k\boldsymbol{\mu})\right) = \boldsymbol{0} \tag{37} $$

where $k = 1 \dots n_{cp}$. These implicit deterministic equations are then solved n_{cp} times with all sources of uncertainty appropriately updated for each solve. The resulting solutions for $\boldsymbol{\mathcal{P}}$ and $\boldsymbol{S}$ are then used to solve for the corresponding expansion coefficients reusing the collocation matrix from (28),

$$ \rho_i^j(t) = A^{\#}\boldsymbol{\mathcal{P}} \tag{38} $$

$$ s_l^j(t) = A^{\#}\boldsymbol{S} \tag{39} $$

where $i = 1 \dots n_{dv}$, $j = 1 \dots n_b$, and $j = 1 \dots n_o$.

4.2.2 Uncertain ODE Systems

To apply gPC to ODE systems, as described by (3)–(5), the independent sources of uncertainty, $\boldsymbol{\theta}(\boldsymbol{\xi})$, may come from system parameters, initial conditions, sensor and actuator noise, or external forcing. The states become dependent uncertain quantities, or $\boldsymbol{\rho}(\boldsymbol{\xi}) = \{\boldsymbol{q}(\boldsymbol{\xi}), \dot{\boldsymbol{q}}(\boldsymbol{\xi}), \ddot{\boldsymbol{q}}(\boldsymbol{\xi}), \boldsymbol{v}(\boldsymbol{\xi}), \dot{\boldsymbol{v}}(\boldsymbol{\xi})\}$. Therefore, each independent and dependent quantity needs to be approximated by (11) or (19), respectively. Substituting these approximations into the Euler-Lagrange EOMs (3) yields,

$$ \begin{aligned} &\boldsymbol{M}\left(\sum_{j=0}^{n_b}\boldsymbol{q}^j\Psi^j, \sum_{j=0}^{p_o}\theta_k^j\psi^j\right)\sum_{j=0}^{n_b}\dot{\boldsymbol{v}}^j\Psi^j + \boldsymbol{C}\left(\sum_{j=0}^{n_b}\boldsymbol{q}^j\Psi^j, \sum_{j=0}^{n_b}\boldsymbol{v}^j\Psi^j, \sum_{j=0}^{p_o}\theta_k^j\,\psi_k^j\right)\sum_{j=0}^{n_b}\boldsymbol{v}^j\Psi^j \\ &\qquad + \boldsymbol{N}\left(\sum_{j=0}^{n_b}\boldsymbol{q}^j\Psi^j, \sum_{j=0}^{n_b}\boldsymbol{v}^j\Psi^j, \sum_{j=0}^{p_o}\theta_k^j\,\psi_k^j\right) \\ &= \boldsymbol{\mathcal{F}}\left(\sum_{j=0}^{n_b}\boldsymbol{q}^j\Psi^j, \sum_{j=0}^{n_b}\boldsymbol{v}^j\Psi^j, \sum_{j=0}^{n_b}\dot{\boldsymbol{v}}^j\Psi^j, \sum_{j=0}^{p_o}\theta_k^j\,\psi_k^j\right) = \sum_{j=0}^{n_b}\boldsymbol{\tau}^j\Psi^j \end{aligned} \tag{40} $$

Equation (4) becomes,

$$ \sum_{j=0}^{n_b}\dot{\boldsymbol{q}}^j\Psi^j = \boldsymbol{H}\left(\sum_{j=0}^{n_b}\boldsymbol{q}^j\Psi^j, \sum_{j=0}^{p_o}\theta_k^j\,\psi_k^j\right)\sum_{j=0}^{n_b}\boldsymbol{v}^j\Psi^j \tag{41} $$

And (5),

$$ \sum_{j=0}^{n_b}\boldsymbol{y}^j\Psi^j = \boldsymbol{\mathcal{O}}\left(\sum_{j=0}^{n_b}\boldsymbol{q}^j\Psi^j, \sum_{j=0}^{n_b}\dot{\boldsymbol{q}}^j\Psi^j, \sum_{j=0}^{p_o}\theta_k^j\,\psi_k^j\right) \tag{42} $$

where, $k = 1 \dots n_p$. Applying the preferred LSCM to (40)–(42) requires defining intermediate variables for the states,

$$ {}_k\mathcal{Q}_i\big(t;\ {}_k\boldsymbol{\mu}\big) = \sum_{j=0}^{n_b} \mathcal{Q}_i^j(t)\Psi^j\left(\ {}_k\boldsymbol{\mu}\right), \qquad i = 1 \dots n_{dv}, k = 0 \dots n_{cp} \tag{43} $$

where $n_{dv} = n_s$, and corresponding intermediate variables for the generalized velocities, uncertainties, inputs and outputs are created. Substituting these intermediate variables along with (26) into (40)–(42) yields,

$$ \boldsymbol{\mathcal{F}}\left(\ {}_k\boldsymbol{\mathcal{Q}}\big(t;\ {}_k\boldsymbol{\mu}\big),\ {}_k\boldsymbol{V}\big(t;\ {}_k\boldsymbol{\mu}\big),\ {}_k\dot{\boldsymbol{V}}\big(t;\ {}_k\boldsymbol{\mu}\big),\ {}_k\boldsymbol{\Theta}\big(t;\ {}_k\boldsymbol{\mu}\big)\right) = \ {}_k\mathbf{T}\big(t;\ {}_k\boldsymbol{\mu}\big) \tag{44} $$
$$ {}_k\dot{\boldsymbol{\mathcal{Q}}}\big(t;\ {}_k\boldsymbol{\mu}\big) = \boldsymbol{H}\left(\ {}_k\boldsymbol{\mathcal{Q}}\big(t;\ {}_k\boldsymbol{\mu}\big),\ {}_k\boldsymbol{\Theta}\big(t;\ {}_k\boldsymbol{\mu}\big)\right)\ {}_k\boldsymbol{V}\big(t;\ {}_k\boldsymbol{\mu}\big) \tag{45} $$
$$ {}_k\boldsymbol{Y}\big(t;\ {}_k\boldsymbol{\mu}\big) = \boldsymbol{\mathcal{O}}\left(\ {}_k\boldsymbol{\mathcal{Q}}\big(t;\ {}_k\boldsymbol{\mu}\big),\ {}_k\dot{\boldsymbol{\mathcal{Q}}}\big(t;\ {}_k\boldsymbol{\mu}\big),\ {}_k\boldsymbol{\Theta}\big(t;\ {}_k\boldsymbol{\mu}\big)\right) \tag{46} $$

where $k = 1 \dots n_{cp}$. The deterministic system (44)–(46) is then solved n_{cp} times with all sources of uncertainty appropriately updated for each solve. The resulting solutions for $\boldsymbol{\mathcal{Q}}$, $\dot{\boldsymbol{\mathcal{Q}}}$, and $\boldsymbol{Y}$ are then used to solve for the corresponding stochastic coefficients reusing the collocation matrix from (28),

$$ q_i^j(t) = A^{\#}\mathcal{Q}_i \tag{47} $$
$$ \dot{q}_i^j(t) = A^{\#}\dot{\mathcal{Q}}_i \tag{48} $$
$$ y_l^j(t) = A^{\#}Y_i \tag{49} $$

where $i = 1 \dots n_s, j = 1 \dots n_b$ and $l = 1 \dots n_o$. Corresponding statistics are subsequently calculated based on (32)–(33).

As previously mentioned, the nonintrusive nature of the LSCM sampling approach is arguably its greatest benefit. Standard ODE solvers may be repeatedly used without changing the base deterministic ODE system. Also, since optimization problems require repeated system solves—which are typically the most expensive portion of the computation in a given optimization step—computational efficiency is of utmost importance. As mentioned, traditional multibody dynamic EOM formulations require computations on the order of the number of dynamic states cubed $O(n_s^3)$. Therefore, applying the LSCM results in $O\big(n_{cp}n_s^3\big)$ complexity which is much less than GPM's $O((n_b n_s)^3)$ complexity.

4.2.3 Benefits and Limitations of gPC

Generalized Polynomial Chaos has been shown to be an efficient computational approach in both the time and frequency domains for large nonlinear multibody systems under the influence of uncertainties with large magnitudes [9, 17]. Its exponential convergence properties make it significantly more efficient than Monte Carlo based methods.

Some of the limitations of gPC based uncertainty quantification are its C^1 continuity requirements and its degrading performance for long simulations. Recent work in the area of multi-element gPC, or ME-gPC, extends gPC's abilities to support discontinuities as well as improves its performance for long simulations [11-16, 40, 41].

5 Motion Planning of Uncertain ODE Systems

5.1 Motivation

This section presents the novel nonlinear programming (NLP) based motion planning framework that treats smooth, uncertain, fully and under-actuated dynamical systems described by ordinary differential equations (ODEs). Uncertainty in multibody dynamical systems comes from various sources, such as: system parameters, initial conditions, sensor and actuator noise, and external forcing. Treatment of uncertainty in design is of paramount practical importance because all real-life systems are affected by it, and poor robustness and suboptimal performance result if it's not accounted for in a given design. System uncertainties are modeled using Generalized Polynomial Chaos (gPC) and are solved quantitatively using a least-square collocation method (LSCM). The computational efficiencies of this approach enable the inclusion of uncertainty statistics in the dynamic optimization process. As such, new design questions related to uncertain dynamical systems can now be answered through the new framework.

Specifically, this section presents the new framework through forward, inverse, and hybrid dynamics formulations. The forward dynamics formulation, applicable to both fully and under-actuated systems, prescribes deterministic actuator inputs which yield uncertain state trajectories. The inverse dynamics formulation, however, is the dual to the forward dynamics formulation and is only applicable to fully-actuated systems; it has prescribed deterministic state trajectories which yield uncertain actuator inputs. The inverse dynamics formulation is more computationally efficient as it is only an algebraic evaluation and completely avoids any numerical integration. Finally, the hybrid dynamics formulation as applicable to under-actuated systems where it leverages the benefits of inverse dynamics for actuated joints and forward dynamics for unactuated joints; it prescribes actuated state and unactuated input trajectories which yield uncertain unactuated states and actuated inputs. The benefits of the ability to quantify uncertainty when planning motion of multibody dynamic systems are illustrated in various optimal motion planning case-studies. The resulting designs determine optimal motion plans—subject to deterministic and statistical constraints—for all possible systems within the probability space.

It's important to point out that the new framework is not dependent on the specific formulation of the dynamical equations of motion (EOMs); formulations such as, Newtonian-Euler, Euler-Lagrange, Hamiltonian, and Geometric methodologies are all applicable. All case-studies, therefore, make use of the analytical Euler-Lagrange EOM formulation.

A brief presentation of the common practice of parameterizing known state and/or actuator input trajectories will be presented first.

5.2 Parameterizing Search Space

One specific type of optimal design problem for multibody dynamic systems is that of planning kinodynamic motions. Fundamentally, the kinodynamic motion planning problem searches for the actuation inputs, $\boldsymbol{\tau}$, and/or joint trajectories, $\{\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}\}$, that minimize J. These trajectories are infinitely dimensioned for continuous dynamics. Even the number of discrete points determined by a numerical solver can be prohibitively large. Therefore, a commonly used technique is to reduce the number of manipulated variables in $\boldsymbol{x}$ by parameterizing the trajectories. There are a number of possible techniques. A brief overview of the Power Series and B-Spline techniques follows.

5.2.1 Power Series

In [69], the authors presented a power series method for parameterizing a curve.

$$y(t) = a_1 + a_2 t + a_2 t^2 + \cdots + a_m t^{m-1} \tag{50}$$

By differentiating (50) twice yields the parameterized velocity and acceleration of the curve. By specifying the initial and terminal conditions of the curve then a designer can solve for the power series coefficients by,

$$\begin{bmatrix} 1 & t_0 & t_0^2 & t_0^3 & t_0^4 & t_0^5 & t_0^6 \\ 0 & 1 & 2t_0 & 3t_0^2 & 4t_0^3 & 5t_0^4 & 6t_0^5 \\ 0 & 0 & 2 & 6t_0 & 12t_0^2 & 20t_0^3 & 30t_0^4 \\ 1 & t_m & t_m^2 & t_m^3 & t_m^4 & t_m^5 & t_m^6 \\ 1 & t_f & t_f^2 & t_f^3 & t_f^4 & t_f^5 & t_f^6 \\ 0 & 1 & 2t_f & 3t_f^2 & 4t_f^3 & 5t_f^4 & 6t_f^5 \\ 0 & 0 & 2 & 6t_f & 12t_f^2 & 20t_f^3 & 30t_f^4 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{bmatrix} = \begin{bmatrix} y(t_0) \\ \dot{y}(t_0) \\ \ddot{y}(t_0) \\ y(t_m) \\ y(t_f) \\ \dot{y}(t_f) \\ \ddot{y}(t_f) \end{bmatrix} \tag{51}$$

where the right-hand-side (RHS) is specified and the vector of a_i coefficients are found by a linear system solve. Notice, that by inserting an intermediate point, $y(t_m)$, yields a sixth-order power series.

Applying this approach in an optimization setting would require including $y(t_m)$ in the list of manipulated variables, $\boldsymbol{x}$. Additional flexibility is obtained by optionally including t_m in $\boldsymbol{x}$ as well.

Simplicity is one of the major benefits of the power series parameterization method. However, two major limitations of this method are the lack of local control and the lack of the ability to bound the curve within an arbitrary magnitude envelope.

5.2.2 B-Spline

B-Splines are arguably the most popular parameterization method [42, 47, 49, 50, 52, 55, 56] in the motion planning literature. In [179], B-Spline curves have the form,

$$\mathbf{B}(u) = \sum_{i=0}^{n_{sp}} \beta^{i,\wp}(u)\boldsymbol{p}^i \tag{52}$$

where there are $(n_{sp}+1)$ control points $\boldsymbol{P} = \{\boldsymbol{p}^0, \dots, \boldsymbol{p}^{n_{sp}}\} \in \mathbb{R}^{n_{sp}+1} \times \mathbb{R}^{n_{dim}}$ with $\boldsymbol{p}^i \in \mathbb{R}^{n_{dim}}$, or $\boldsymbol{p}^{i,j}$ is the jth element of the ith control point; $m+1$ non-decreasing knots $u^0 \leq \cdots \leq u^m \in \mathbb{R}$; and $(n_{sp}+1)$ basis $\beta^{i,\wp}(u)$ of degree of $\wp$; where the relation $m = n_{sp} + \wp + 1$ must be maintained.

The knot span $[u^0, u^m)$ can be defined to correspond with the time of a motion plan's trajectory; where $u^0 = t_0$ and $u^m = t_f$, or $\beta^{i,\wp}(u) = \beta^{i,\wp}(t)$. Therefore, the curve $\mathbf{B}(u) = \mathbf{B}(t)$ is defined from $[t_0, t_m)$.

Basis functions, $\beta^{i,\wp}(u)$, can be created recursively by the *Cox-de Boor recursion formula*.

$$\begin{gathered}\beta^{i,0}(u) = \begin{cases}1 & \text{if } u^i \leq u < u^{i+1} \\ 0 & \text{otherwise}\end{cases} \\ \beta^{i,\wp}(u) = \frac{u - u^i}{u^{i+\wp} - u^i}\beta^{i,\wp-1}(u) + \frac{u^{i+\wp+1} - u}{u^{i+\wp+1} - u^{i+1}}\beta^{i+1,\wp-1}(u)\end{gathered} \tag{53}$$

Also, a *clamped* B-spline has $(\wp + 1)$ repeated knots at the extremes of the knot list. The clamping allows one to force the curve to be tangent to the first and last control point legs at the first and last control points. Meaning, $\mathbf{B}(u^0) = \boldsymbol{p}^0$ and $\mathbf{B}(u^m) = \boldsymbol{p}^{n_{sp}}$. This enables one to specify the initial and terminal conditions for the curve by the initial and final control points. The remaining interior control points specify the shape of the curve.

5.2.2.1 Derivatives of B-Splines

Derivatives of a B-Spline are also B-Splines, therefore, let $\mathbf{B}'(u) = \frac{\partial \mathbf{B}(u)}{\partial u}$ represent the first derivative of $\mathbf{B}(u)$. With another slight abuse of Lagrange's derivative notation, let the control points for $\mathbf{B}'(u)$ be defined as $\boldsymbol{P}' = \{\boldsymbol{p}'^0, \dots, \boldsymbol{p}'^{n_{sp}-1}\}$. Unlike $\boldsymbol{P}$, the values for $\boldsymbol{P}'$ are predetermined through the following recursive relation,

$$\boldsymbol{p}'^i = \frac{\wp}{u^{i+\wp+1} - u^{i+1}}\left(\boldsymbol{p}^{i+1} - \boldsymbol{p}^i\right) \tag{54}$$

resulting $n_{sp} - 1$ inherited control points; or, $\boldsymbol{P}' \in \mathbb{R}^{n_{sp}-1} \times \mathbb{R}^{n_{dim}}$. The corresponding $n_{sp} - 1$ basis functions, $\beta^{i,\wp-1}(u)$, are of degree $\wp - 1$ and are also calculated using (53).

Additionally, all derivative B-Splines inherit their knot vector from their *parent* B-Spline. However, only a subset of the original knot vector is used. Meaning, the knot vector for a derivative, $\boldsymbol{u}'$, is updated by removing the first and last knot from the original knot vector, $\boldsymbol{u}$.

$$\boldsymbol{u}' = \{u^1 \leq \cdots \leq u^{m-1}\} \subset \boldsymbol{u} \tag{55}$$

These recursive relations for control points, basis, and knot vectors also apply for higher-order derivatives. Therefore, by defining $\boldsymbol{P}$ for $\mathbf{B}(u)$ all of its derivatives supported by the original degree $\wp$, control points and knots are automatically defined.

To illustrate, given $\mathbf{B}(u)$ defined in (52), the first and second derivative curves are defined by,

$$\mathbf{B}'(u') = \sum_{i=0}^{n_{sp}-1} \beta^{i,\wp-1}(u')\boldsymbol{p}'^{i} \tag{56}$$

$$\mathbf{B}''(u'') = \sum_{i=0}^{n_{sp}-2} \beta^{i,\wp-2}(u'')\boldsymbol{p}''^{i} \tag{57}$$

Therefore, in order to specify the initial and/or terminal conditions of a derivative *clamped* B-Spline, the slope of the first/last leg of its *parent's* control points must match the value for the initial/final condition for the derivative. These are determined from (54).

5.2.2.2 Use of B-Splines in Motion Planning Problems

In an optimization setting, if the design was searching for a trajectory $q(t)$ that minimized a given J, significant efficiencies are gained by parameterizing the trajectory as,

$$q(\boldsymbol{P}, t) = \sum_{i=0}^{n_{sp}} \beta^{i,\wp}(u)\boldsymbol{p}^{i} \tag{58}$$

Therefore, instead of including $q(t)$ and its derivatives in the list of manipulated variables, $\boldsymbol{x} = \{q(t), \dot{q}(t), \ddot{q}(t), \dots\}$, the finite-dimensional list of B-Spline control points are used, $\boldsymbol{x} = \{\boldsymbol{P}, \dots\}$. In fact, given the previous discussion about initial and terminal conditions of *clamped* B-Splines, if either the initial or terminal conditions of the trajectory are known then only the remaining interior set of control points are included in $\boldsymbol{x}$. So, in the case that both initial and terminal conditions are specified, $\boldsymbol{x} = \{\boldsymbol{P}^{i}\},\ i = 1 \dots (n_{sp} - 1)$. Likewise, if initial/terminal conditions are prescribed for the corresponding derivative spline $\dot{q}(\boldsymbol{P}', t)$, then the set of free interior control points shrinks by two; meaning, for the case that both initial and terminal derivatives are specified, then the resulting list of manipulated variables would be $\boldsymbol{x} = \{\boldsymbol{P}^{i}\},\ i = 2 \dots (n_{sp} - 2)$.

Additionally, when applying a B-Spline to an optimization problem, a designer needs to decide if both the *knots* and *control points* will be included in the list of manipulated variables, $\boldsymbol{x}$. The appropriate subset of *control* points must be included; however, including the *knots* results in a parameterized curve with more degrees of freedom, but, the cost of more manipulated variables in the optimization search. Another drawback of adding the *knots* is they must be non-decreasing. Therefore, additional bounding constraints on the individual *knots* need to be included in $\boldsymbol{C}$. If a reduction of freedom in the curve is acceptable then predefined *knot* values can be used and only the appropriate *control points* are added to $\boldsymbol{x}$. The later scenario is more commonly used in the literature. All subsequent formulations will assume only the *control points* are included in $\boldsymbol{x}$.

In conclusion, B-Splines are more complicated in concept and in implementation than Power Series, however, they have added benefits. First, they provide local control of the curve; changing a single control point does not cause unexpected variations in the curve in seemingly unrelated regions. Also, the bounds of the curve are easily controlled by bounding the control points, $\boldsymbol{P}$, since the curve never exceeds the convex hull of the control points.

The remainder of this work will assume that all parameterized trajectories are B-Spline curves.

5.3 Motion Planning of Deterministic Fully-Actuated and Under-Actuated systems

The task of dynamic system motion planning is a well studied topic; it aims to determine either a state or input trajectory—or an appropriate combination—to realize some prescribed motion objective. Treatment of fully and under-actuated systems presents multiple methodologies for formulating the governing dynamics. The *forward dynamics* formulation, applicable to both fully and under-actuated systems, prescribes actuator inputs which yield state trajectories through numerical integration. The *inverse dynamics* formulation is the dual to the *forward dynamics* formulation and is only applicable to fully-actuated systems; it has prescribed state trajectories which yield actuator inputs. The *inverse dynamics* formulation is more computationally efficient as it is only an algebraic evaluation and completely avoids any numerical integration. Finally, the *hybrid dynamics* formulation is applicable to under-actuated systems and leverages the benefits of *inverse dynamics* for actuated joints and relies on *forward dynamics* for unactuated joints; it prescribes actuated state and unactuated input trajectories to determine unactuated states through numerical integration and actuated inputs through algebraic evaluations. Partitioning the system states and inputs between actuated and unactuated joints in the following manner, $\boldsymbol{q} = \{\, {}^{a}\boldsymbol{q},\ {}^{u}\boldsymbol{q}\}$ and $\boldsymbol{\tau} = \{\, {}^{a}\boldsymbol{\tau},\ {}^{u}\boldsymbol{\tau}\}$, facilitates the illustration of what quantities are known versus unknown when using these formulations of the system's dynamics (see Table 5.1).

Table 5.1—Knowns vs Unknowns Dynamic Properties

Formulation	**Known**	**Unknown**
Forward	$\boldsymbol{\tau}$	$\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \boldsymbol{v}, \dot{\boldsymbol{v}}$
Inverse	$\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \boldsymbol{v}, \dot{\boldsymbol{v}}$	$\boldsymbol{\tau}$
Hybrid	${}^{a}\boldsymbol{q},\ {}^{a}\dot{\boldsymbol{q}},\ {}^{a}\ddot{\boldsymbol{q}},\ {}^{a}\boldsymbol{v},\ {}^{a}\dot{\boldsymbol{v}},\ {}^{u}\boldsymbol{\tau}$	${}^{u}\boldsymbol{q},\ {}^{u}\dot{\boldsymbol{q}},\ {}^{u}\ddot{\boldsymbol{q}},\ {}^{u}\boldsymbol{v},\ {}^{u}\dot{\boldsymbol{v}},\ {}^{a}\boldsymbol{\tau}$

Regardless of which dynamics formulation is selected, a common motion planning practice is to approximate infinite dimensional *known* trajectories by a finite dimensional parameterization [51]. This paper parameterizes all *known* trajectories with B-Splines. For example, the parameterization of $\boldsymbol{q}$ takes the form,

$$\boldsymbol{q}(\boldsymbol{P}, u) = \sum_{i=0}^{n_{sp}} \beta^{i,\mathscr{p}-1}(u)\boldsymbol{p}^{i} \tag{59}$$

and a similar expansion is given for $\boldsymbol{\tau}(\boldsymbol{P}, u)$. There are $\left(n_{sp}+1\right)$ control points $\boldsymbol{P} = \{\boldsymbol{p}^{0}, \dots, \boldsymbol{p}^{n_{sp}}\} \in \mathbb{R}^{n_{sp}+1} \times \mathbb{R}^{n_{dim}}$ with $\boldsymbol{p}^{i} \in \mathbb{R}^{n_{dim}}$, where $\boldsymbol{p}^{i,j}$ is the *j*th element of the *i*th control point; $m+1$ non-decreasing knots $u^{0} \leq \cdots \leq u^{m} \in \mathbb{R}$; and $\left(n_{sp}+1\right)$ basis $\beta^{i,\mathscr{p}}(u)$ of degree of $\mathscr{p}$; and the relation $m = n_{sp} + \mathscr{p} + 1$ must be maintained.

Basis functions, $\beta^{i,\mathscr{p}}(u)$, can be created recursively by the *Cox-de Boor recursion formula*.

$$\begin{gathered} \beta^{i,0}(u) = \begin{cases} 1 & \text{if } u^{i} \leq u < u^{i+1} \\ 0 & \text{otherwise} \end{cases} \\ \beta^{i,\mathscr{p}}(u) = \frac{u-u^{i}}{u^{i+\mathscr{p}}-u^{i}}\beta^{i,\mathscr{p}-1}(u) + \frac{u^{i+\mathscr{p}+1}-u}{u^{i+\mathscr{p}+1}-u^{i+1}} \end{gathered} \tag{60}$$

Also, a *clamped* B-spline has $(\mathscr{p}+1)$ repeated knots at the extremes of the knot list. The clamping allows one to force the curve to be tangent to the first and last control point legs at the first and last control points. Meaning, $\boldsymbol{\tau}(\boldsymbol{P}, u^{0}) = \boldsymbol{p}^{0}$ and $\boldsymbol{\tau}(\boldsymbol{P}, u^{m}) = \boldsymbol{p}^{n_{sp}}$. This enables one to specify the initial and terminal conditions for the curve by the initial and final control points. The remaining interior control points specify the shape of the curve.

Derivatives of B-Spline functions are also B-Splines. Let $\mathbf{q}'(\boldsymbol{P}, u) = \frac{\partial \mathbf{q}(u)}{\partial u}$ represent the first derivative of $\mathbf{q}(\boldsymbol{P}, u)$. With a slight abuse of Lagrange's derivative notation, let the control points for $\mathbf{q}'(\boldsymbol{P}, u)$ be defined as $\boldsymbol{P}' = \{\boldsymbol{p}'^{0}, \dots, \boldsymbol{p}'^{n_{sp}-1}\}$. Unlike $\boldsymbol{P}$, the values of $\boldsymbol{P}'$ are predetermined through the following recursive relation,

$$\boldsymbol{p}'^{i} = \frac{\wp}{u^{i+\wp+1} - u^{i+1}} \left(\boldsymbol{p}^{i+1} - \boldsymbol{p}^{i} \right) \tag{61}$$

which gives the $n_{sp} - 1$ inherited control points; or, $\boldsymbol{P}' \in \mathbb{R}^{n_{sp}-1} \times \mathbb{R}^{n_{dim}}$. The corresponding $n_{sp} - 1$ basis functions, $\beta^{i,\wp-1}(u)$, are of degree $\wp - 1$ and are also calculated using (53).

Additionally, all derivative B-Splines inherit their knot vector from their *parent* B-Spline. However, only a subset of the original knot vector is used. Meaning, the knot vector for a derivative, $\boldsymbol{u}'$, is updated by removing the first and last knot from the original knot vector, $\boldsymbol{u}$,

$$\boldsymbol{u}' = \{u^{1} \leq \cdots \leq u^{m-1}\} \subset \boldsymbol{u}. \tag{62}$$

These recursive relations for control points, basis, and knot vectors also apply for higher-order derivatives. Therefore, by defining $\boldsymbol{P}$ for $\boldsymbol{q}(\boldsymbol{P}, u)$, all of its derivatives supported by the original degree $\wp$, control points, and knots, are automatically defined [180].

To illustrate, given $\mathbf{q}(u)$ defined in (59), the first and second derivative curves are defined by,

$$\mathbf{q}'(u') = \sum_{i=0}^{n_{sp}-1} \beta^{i,\wp-1}(u')\boldsymbol{p}'^{i} \tag{63}$$

$$\mathbf{q}''(u'') = \sum_{i=0}^{n_{sp}-2} \beta^{i,\wp-2}(u'')\boldsymbol{p}''^{i} \tag{64}$$

Therefore, in order to specify the initial and/or terminal conditions of a derivative *clamped* B-Spline, the slope of the first/last leg of its *parent's* control points must match the value for the initial/final condition for the derivative. These are determined from (54).

In a motion planning setting, the knot span $[u^{0}, u^{m})$ can be defined to correspond to the time of a motion plan's trajectory; where $u^{0} = t_{0}$ and $u^{m} = t_{f}$, or $\beta^{i,\wp}(u) = \beta^{i,\wp}(t)$. Therefore, the curves $\boldsymbol{q}(\boldsymbol{P}, u) = \boldsymbol{q}(\boldsymbol{P}, t)$ and $\boldsymbol{\tau}(\boldsymbol{P}, u) = \boldsymbol{\tau}(\boldsymbol{P}, t)$ are defined from $[t_{0}, t_{f})$.

The generalized velocities and accelerations, $\boldsymbol{v}(\boldsymbol{P}',t)$ and $\dot{\boldsymbol{v}}(\boldsymbol{P}'',t)$, respectively, may be determined by differentiating (4) twice, yielding,

$$\ddot{\boldsymbol{q}}(\boldsymbol{P},t)=\boldsymbol{H}(\boldsymbol{q}(\boldsymbol{P},t),\boldsymbol{\theta})\dot{\boldsymbol{v}}(\boldsymbol{P}'',t)+\boldsymbol{v}(\boldsymbol{P}',t)\left(\frac{\partial\boldsymbol{H}}{\partial t}+\frac{\partial\boldsymbol{H}}{\partial\boldsymbol{q}}\frac{\partial\boldsymbol{q}}{\partial t}+\frac{\partial\boldsymbol{H}}{\partial\boldsymbol{\theta}}\frac{\partial\boldsymbol{\theta}}{\partial t}\right) \tag{65}$$

Solving (4) for $\boldsymbol{v}(\boldsymbol{P}',t)$ and (65) for $\dot{\boldsymbol{v}}(\boldsymbol{P}'',t)$ yields,

$$\boldsymbol{v}(\boldsymbol{P}',t)=\big(\boldsymbol{H}(\boldsymbol{q}(\boldsymbol{P},t),\boldsymbol{\theta})\big)^{-1}\dot{\boldsymbol{q}}(\boldsymbol{P}',t) \tag{66}$$

$$\dot{\boldsymbol{v}}(\boldsymbol{P}'',t)=\big(\boldsymbol{H}(\boldsymbol{q}(\boldsymbol{P},t),\boldsymbol{\theta})\big)^{-1}\left(\ddot{\boldsymbol{q}}(\boldsymbol{P}'',t)-\boldsymbol{v}(\boldsymbol{P}',t)\left(\frac{\partial\boldsymbol{H}}{\partial t}+\frac{\partial\boldsymbol{H}}{\partial\boldsymbol{q}}\frac{\partial\boldsymbol{q}}{\partial t}+\frac{\partial\boldsymbol{H}}{\partial\boldsymbol{\theta}}\frac{\partial\boldsymbol{\theta}}{\partial t}\right)\right) \tag{67}$$

The parameterizations (59), (65)–(67) are equally applicable to appropriate actuated and unactuated subsets.

Once all known trajectories are parameterized the EOMs take on the form,

$$\text{Forward:}\quad \boldsymbol{\mathcal{F}}(\boldsymbol{q}(\boldsymbol{P}),\boldsymbol{v}(\boldsymbol{P}'),\dot{\boldsymbol{v}}(\boldsymbol{P}''),\boldsymbol{\theta})=\boldsymbol{\tau} \tag{68}$$

$$\text{Inverse:}\quad \boldsymbol{\tau}=\boldsymbol{\mathcal{F}}(\boldsymbol{q}(\boldsymbol{P}),\boldsymbol{v}(\boldsymbol{P}'),\dot{\boldsymbol{v}}(\boldsymbol{P}''),\boldsymbol{\theta}) \tag{69}$$

$$\text{Hybrid:}\quad \begin{pmatrix}{}^{u}\dot{\boldsymbol{v}}\\ {}^{a}\boldsymbol{\tau}\end{pmatrix}=\boldsymbol{\mathcal{G}}({}^{a}\boldsymbol{q}(\boldsymbol{P}),\ {}^{a}\boldsymbol{v}(\boldsymbol{P}'),\ {}^{a}\dot{\boldsymbol{v}}(\boldsymbol{P}''),\ {}^{u}\boldsymbol{\tau}(\boldsymbol{P}),\boldsymbol{\theta}) \tag{70}$$

where the time dependence has been dropped again for notational convenience.

In the *hybrid dynamics* case, it is worth mentioning that the unactuated input wrenches, ${}^{u}\boldsymbol{\tau}$, represent joint constraint forces. Depending on the formulation used to determine the EOMS (e.g., analytic versus recursive methods), then ${}^{u}\boldsymbol{\tau}$ may be implicitly known once $\{{}^{a}\boldsymbol{q}(\boldsymbol{P}),\ {}^{a}\boldsymbol{v}(\boldsymbol{P}),\ {}^{a}\dot{\boldsymbol{v}}(\boldsymbol{P})\}$ are specified. In such a formulation (70) reduces to,

$$\begin{pmatrix}{}^{u}\dot{\boldsymbol{v}}\\ {}^{a}\boldsymbol{\tau}\end{pmatrix}=\boldsymbol{\mathcal{G}}({}^{a}\boldsymbol{q}(\boldsymbol{P}),\ {}^{a}\boldsymbol{v}(\boldsymbol{P}),\ {}^{a}\dot{\boldsymbol{v}}(\boldsymbol{P}),\boldsymbol{\theta}) \tag{71}$$

Once (68)–(71) are determined then the NLP-based deterministic motion planning problem may be formulated as,

Forward Dynamics NLP Formulation:

$$\begin{aligned}\min_{x=\{\boldsymbol{P}\}}\quad & \mathrm{J}\\ \text{s.t.}\quad & \text{Forward dynamics}\\ & \boldsymbol{\mathcal{F}}(\boldsymbol{q},\boldsymbol{v},\dot{\boldsymbol{v}},\boldsymbol{\theta})=\boldsymbol{\tau}(\boldsymbol{P})\\ & \text{Kinematics}\\ & \dot{\boldsymbol{q}}=\boldsymbol{H}(\boldsymbol{q},\boldsymbol{\theta})\boldsymbol{v}\end{aligned} \tag{72}$$

Outputs

$$\boldsymbol{y} = \boldsymbol{\mathcal{O}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\theta})$$

Constraints

$$\boldsymbol{\mathcal{C}}(\boldsymbol{y}, \boldsymbol{\tau}, \boldsymbol{\theta}) \leq \boldsymbol{0}$$

Hard ICs & TCs Conditions

$$\boldsymbol{q}(0) = \boldsymbol{q}_0$$
$$\dot{\boldsymbol{q}}(0) = \dot{\boldsymbol{q}}_0,$$
$$\boldsymbol{q}\left(t_f\right) = \boldsymbol{q}_{t_f}$$
$$\dot{\boldsymbol{q}}\left(t_f\right) = \dot{\boldsymbol{q}}_{t_f}$$

Inverse Dynamic NLP Formulation:

$$\min_{\boldsymbol{x}=\{\boldsymbol{P}\}} \quad \mathrm{J}$$

$\mathrm{s.t.}$

Kinematics

$$\boldsymbol{v}(\boldsymbol{P}') = \left(\boldsymbol{H}(\boldsymbol{q}(\boldsymbol{P}), \boldsymbol{\theta})\right)^{-\mathbf{1}} \dot{\boldsymbol{q}}(\boldsymbol{P}')$$

$$\dot{\boldsymbol{v}}(\boldsymbol{P}'') = \left(\boldsymbol{H}(\boldsymbol{q}(\boldsymbol{P}), \boldsymbol{\theta})\right)^{-\mathbf{1}} \left(\ddot{\boldsymbol{q}}(\boldsymbol{P}'') - \boldsymbol{v}(\boldsymbol{P}') \left(\frac{\partial \boldsymbol{H}}{\partial t} + \frac{\partial \boldsymbol{H}}{\partial \boldsymbol{q}} \frac{\partial \boldsymbol{q}}{\partial t} + \frac{\partial \boldsymbol{H}}{\partial \boldsymbol{\theta}} \frac{\partial \boldsymbol{\theta}}{\partial t}\right)\right)$$

Inverse Dynamics

$$\boldsymbol{\tau} \ = \boldsymbol{\mathcal{F}}(\boldsymbol{q}(\boldsymbol{P}), \boldsymbol{v}(\boldsymbol{P}'), \dot{\boldsymbol{v}}(\boldsymbol{P}''), \boldsymbol{\theta}) \tag{73}$$

Outputs

$$\boldsymbol{y} = \boldsymbol{\mathcal{O}}(\boldsymbol{q}(\boldsymbol{P}), \dot{\boldsymbol{q}}(\boldsymbol{P}'), \boldsymbol{\theta})$$

Constraints

$$\boldsymbol{\mathcal{C}}(\boldsymbol{y}, \boldsymbol{\tau}, \boldsymbol{\theta}) \leq \boldsymbol{0}$$

Hard ICs & TCs Conditions

$$\boldsymbol{q}(0) = \boldsymbol{P}^0 = \boldsymbol{q}_0$$
$$\dot{\boldsymbol{q}}(0) = \boldsymbol{P}'^0 = \dot{\boldsymbol{q}}_0$$
$$\boldsymbol{q}\left(t_f\right) = \boldsymbol{P}^{n_{sp}} = \boldsymbol{q}_{t_f}$$
$$\dot{\boldsymbol{q}}\left(t_f\right) = \boldsymbol{P}'^{n_{sp}-1} = \dot{\boldsymbol{q}}_{t_f}$$

Hybrid Dynamics NLP Formulation:

$$\min_{\boldsymbol{x}=\{\boldsymbol{P}\}} \quad \mathrm{J}$$

$\mathrm{s.t.}$

Actuated kinematics

$${}^{\boldsymbol{a}}\boldsymbol{v}(\boldsymbol{P}') = \left(\boldsymbol{H}({}^{\boldsymbol{a}}\boldsymbol{q}(\boldsymbol{P}), \boldsymbol{\theta})\right)^{-\mathbf{1}} \ {}^{\boldsymbol{a}}\dot{\boldsymbol{q}}(\boldsymbol{P}')$$

$${}^{\boldsymbol{a}}\dot{\boldsymbol{v}}(\boldsymbol{P}'') = \left(\boldsymbol{H}({}^{\boldsymbol{a}}\boldsymbol{q}(\boldsymbol{P}), \boldsymbol{\theta})\right)^{-\mathbf{1}} \left({}^{\boldsymbol{a}}\ddot{\boldsymbol{q}}(\boldsymbol{P}'') - {}^{\boldsymbol{a}}\boldsymbol{v}(\boldsymbol{P}') \left(\frac{\partial \boldsymbol{H}}{\partial t} + \frac{\partial \boldsymbol{H}}{\partial \boldsymbol{q}} \frac{\partial \boldsymbol{q}}{\partial t} + \frac{\partial \boldsymbol{H}}{\partial \boldsymbol{\theta}} \frac{\partial \boldsymbol{\theta}}{\partial t}\right)\right)$$

Hybrid dynamics

$$\begin{pmatrix} {}^{\boldsymbol{u}}\dot{\boldsymbol{v}} \\ {}^{\boldsymbol{a}}\boldsymbol{\tau} \end{pmatrix} = \boldsymbol{\mathcal{G}}({}^{\boldsymbol{a}}\boldsymbol{q}(\boldsymbol{P}), {}^{\boldsymbol{a}}\boldsymbol{v}(\boldsymbol{P}'), {}^{\boldsymbol{a}}\dot{\boldsymbol{v}}(\boldsymbol{P}''), {}^{\boldsymbol{u}}\boldsymbol{\tau}(\boldsymbol{P}), \boldsymbol{\theta}) \tag{74}$$

Unactuated kinematics

$${}^{\boldsymbol{u}}\dot{\boldsymbol{q}} = {}^{\boldsymbol{u}}\boldsymbol{H}({}^{\boldsymbol{u}}\boldsymbol{q}, \boldsymbol{\theta})\, {}^{\boldsymbol{u}}\boldsymbol{v}$$

Outputs

$$\boldsymbol{y} = \boldsymbol{\mathcal{O}}(\boldsymbol{q}(\boldsymbol{P}), \dot{\boldsymbol{q}}(\boldsymbol{P}'), \boldsymbol{\theta})$$

Constraints

$$\boldsymbol{\mathcal{C}}(\boldsymbol{y}, \boldsymbol{\tau}, \boldsymbol{\theta}) \leq \boldsymbol{0}$$

Hard actuated ICs & TCs Conditions

$${}^{\boldsymbol{a}}\boldsymbol{q}(0) = {}^{\boldsymbol{a}}\boldsymbol{P}^0 = {}^{\boldsymbol{a}}\boldsymbol{q}_0$$

$$
\begin{aligned}
&{}^{\boldsymbol{a}}\dot{\boldsymbol{q}}(0) = {}^{\boldsymbol{a}}\boldsymbol{P}^{0} = {}^{\boldsymbol{a}}\dot{\boldsymbol{q}}_0\\
&{}^{\boldsymbol{a}}\boldsymbol{q}(t_f) = {}^{\boldsymbol{a}}\boldsymbol{P}^{n_{sp}} = {}^{\boldsymbol{a}}\boldsymbol{q}_{t_f}\\
&{}^{\boldsymbol{a}}\dot{\boldsymbol{q}}(t_f) = {}^{\boldsymbol{a}}\boldsymbol{P}^{n_{sp}-1} = {}^{\boldsymbol{a}}\dot{\boldsymbol{q}}_{t_f}
\end{aligned}
$$

Hard unactuated ICs & TCs Conditions

$$
\begin{aligned}
&{}^{\boldsymbol{u}}\boldsymbol{q}(0) = {}^{\boldsymbol{u}}\boldsymbol{q}_0\\
&{}^{\boldsymbol{u}}\dot{\boldsymbol{q}}(0) = {}^{\boldsymbol{u}}\dot{\boldsymbol{q}}_0\\
&{}^{\boldsymbol{u}}\boldsymbol{q}(t_f) = {}^{\boldsymbol{u}}\boldsymbol{q}_{t_f}\\
&{}^{\boldsymbol{u}}\dot{\boldsymbol{q}}(t_f) = {}^{\boldsymbol{u}}\dot{\boldsymbol{q}}_{t_f}
\end{aligned}
$$

Equations (72)–(74) seek to find the control points $\boldsymbol{P}$ that minimize some prescribed objective function, J, while being subject to the dynamic constraints defined in one of (68)–(71). Additional constraints may also be defined; for example, maximum/minimum actuator and system parameter limits or physical system geometric limits can be represented as inequality relations, $\boldsymbol{C}(\boldsymbol{y},\boldsymbol{\tau},\boldsymbol{\theta}) \leq \boldsymbol{0}$. In the *hybrid dynamics* NLP formulation, equation (74) explicitly differentiates between the initial conditions (ICs) and terminal conditions (TCs) for the actuated and unactuated states. All actuated ICs and TCs are determined by corresponding control points in $\boldsymbol{P}$ and all unactuated ICs and TCs are freely defined. Also, equation (74) explicitly presents the TCs *hard* constraints; using the techniques discussed in Section 3.4 they may be defined as *soft* constraints.

The literature contains a variety of objective function definitions for J when used in a motion planning setting. Some commonly defined objective functions are,

$$\boldsymbol{J_{D1}} = \boldsymbol{t_f} \tag{75}$$

$$\mathrm{J_{D2}} = \sum_{i=1}^{n_i} \int_{t_0=0}^{t_f} {\tau_i}^2(t)\, dt \tag{76}$$

$$\mathrm{J_{D3}} = \sum_{i=1}^{n_i} \int_{t_0=0}^{t_f} |\tau_i(t)\dot{q}_{\mathrm{i}}(t)|\, dt \tag{77}$$

$$\mathrm{J_{D4}} = \sum_{i=1}^{n_i} \int_{t_0=0}^{t_f} {\dot{\tau}_i}^2(t)\, dt \tag{78}$$

where (75) represents a *time optimal* design, (76) minimizes the effort, (77) the power, and (78) the jerk.

The solutions to (72)–(74) produce optimal motion plans under the assumption that all system properties are known (i.e., (68)–(71) are completely deterministic). The primary contribution of this work is the presentation of variants of (72)–(74) that allows (68)–(71) to contain uncertainties of diverse types (e.g., parameters, initial conditions, sensor/actuator noise, or forcing functions). The following section will briefly introduce Generalized Polynomial Chaos (gPC) which is used to model the uncertainties and to quantify the resulting uncertain system states and inputs.

5.4 Motion Planning of Uncertain Fully-Actuated and Under-Actuated systems

The deterministic motion planning formulations itemized in equations (72)–(74) do not have the ability to account for uncertainties that are inevitably present in a system. The primary contribution of this paper is the development of a new NLP-based framework that, unlike (72)–(74) in Section 5.3, directly treats system uncertainties during the motion planning process. The formulations based on *forward*, *inverse*, and *hybrid dynamics* are,

Forward Dynamics NLP Formulation:

$$\begin{aligned}
\min_{\boldsymbol{x}=\{\boldsymbol{P}\}} \quad & \mathrm{J} \\
\mathrm{s.t.} \quad & \text{Forward dynamics} \\
& \boldsymbol{\mathcal{F}}\big(\boldsymbol{q}(\boldsymbol{\xi}), \boldsymbol{v}(\boldsymbol{\xi}), \dot{\boldsymbol{v}}(\boldsymbol{\xi}), \boldsymbol{\theta}(\boldsymbol{\xi})\big) = \boldsymbol{\tau}(\boldsymbol{P}) \\
& \text{Kinematics} \\
& \dot{\boldsymbol{q}}(\boldsymbol{\xi}) = \boldsymbol{H}\big(\boldsymbol{q}(\boldsymbol{\xi}), \boldsymbol{\theta}(\boldsymbol{\xi})\big)\boldsymbol{v}(\boldsymbol{\xi}) \\
& \text{Outputs} \\
& \boldsymbol{y}(\boldsymbol{\xi}) = \boldsymbol{\mathcal{O}}\big(\boldsymbol{q}(\boldsymbol{\xi}), \dot{\boldsymbol{q}}(\boldsymbol{\xi}), \boldsymbol{\theta}(\boldsymbol{\xi})\big) \\
& \text{Constraints} \\
& \boldsymbol{\mathcal{C}}(\boldsymbol{y}(\boldsymbol{\xi}), \boldsymbol{\theta}(\boldsymbol{\xi}), \boldsymbol{\tau}(\boldsymbol{P})) \leq \boldsymbol{0} \\
& \text{Hard ICs \& TCs Conditions} \\
& \boldsymbol{q}(0; \boldsymbol{\xi}) = \boldsymbol{q}_0 \\
& \dot{\boldsymbol{q}}(0; \boldsymbol{\xi}) = \dot{\boldsymbol{q}}_0 \\
& \boldsymbol{q}\big(t_f; \boldsymbol{\xi}\big) = \boldsymbol{q}_{t_f} \\
& \dot{\boldsymbol{q}}\big(t_f; \boldsymbol{\xi}\big) = \dot{\boldsymbol{q}}_{t_f}
\end{aligned} \tag{79}$$

Inverse Dynamics NLP Formulation:

$$\begin{aligned}
\min_{\boldsymbol{x}=\{\boldsymbol{P}\}} \quad & \mathrm{J} \\
\mathrm{s.t.} \quad & \text{Kinematics} \\
& \boldsymbol{v}(\boldsymbol{P}') = \big(\boldsymbol{H}(\boldsymbol{q}(\boldsymbol{P}), \boldsymbol{\theta})\big)^{-\mathbf{1}} \dot{\boldsymbol{q}}(\boldsymbol{P}')
\end{aligned} \tag{80}$$

$$\dot{\boldsymbol{v}}(\boldsymbol{P}'') = \left(\boldsymbol{H}(\boldsymbol{q}(\boldsymbol{P}),\boldsymbol{\theta})\right)^{-\mathbf{1}} \left(\ddot{\boldsymbol{q}}(\boldsymbol{P}'') - \boldsymbol{v}(\boldsymbol{P}')\left(\frac{\partial \boldsymbol{H}}{\partial t} + \frac{\partial \boldsymbol{H}}{\partial \boldsymbol{q}}\frac{\partial \boldsymbol{q}}{\partial t} + \frac{\partial \boldsymbol{H}}{\partial \boldsymbol{\theta}}\frac{\partial \boldsymbol{\theta}}{\partial t}\right)\right)$$

Inverse dynamics

$$\boldsymbol{\tau}(\boldsymbol{\xi}) \ = \boldsymbol{\mathcal{F}}\left(\boldsymbol{q}(\boldsymbol{P}), \boldsymbol{v}(\boldsymbol{P}'), \dot{\boldsymbol{v}}(\boldsymbol{P}''), \boldsymbol{\theta}(\boldsymbol{\xi})\right)$$

Outputs

$$\boldsymbol{y}(\boldsymbol{\xi}) = \boldsymbol{\mathcal{O}}\left(\boldsymbol{q}(\boldsymbol{P}), \dot{\boldsymbol{q}}(\boldsymbol{P}'), \boldsymbol{\theta}(\boldsymbol{\xi})\right)$$

Constraints

$$\boldsymbol{\mathcal{C}}(\boldsymbol{y}(\boldsymbol{\xi}), \boldsymbol{\theta}(\boldsymbol{\xi}), \boldsymbol{\tau}(\boldsymbol{\xi})) \leq \mathbf{0}$$

Hard ICs & TCs Conditions

$$\boldsymbol{q}(0) = \boldsymbol{P}^0 = \boldsymbol{q}_0$$

$$\dot{\boldsymbol{q}}(0) = \boldsymbol{P}'^0 = \dot{\boldsymbol{q}}_0$$

$$\boldsymbol{q}\left(t_f\right) = \boldsymbol{P}^{n_{sp}} = \boldsymbol{q}_{t_f}$$

$$\dot{\boldsymbol{q}}\left(t_f\right) = \boldsymbol{P}'^{n_{sp}-1} = \dot{\boldsymbol{q}}_{t_f}$$

Hybrid Dynamics NLP Formulation:

$$\min_{\boldsymbol{x}=\{\boldsymbol{P}\}} \ \mathrm{J}$$

s.t. **Actuated Kinematics**

$${}^{a}\boldsymbol{v}(\boldsymbol{P}') = \left(\boldsymbol{H}({}^{a}\boldsymbol{q}(\boldsymbol{P}),\boldsymbol{\theta})\right)^{-\mathbf{1}} \ {}^{a}\dot{\boldsymbol{q}}(\boldsymbol{P}')$$

$${}^{a}\dot{\boldsymbol{v}}(\boldsymbol{P}'') = \left(\boldsymbol{H}({}^{a}\boldsymbol{q}(\boldsymbol{P}),\boldsymbol{\theta})\right)^{-\mathbf{1}} \left({}^{a}\ddot{\boldsymbol{q}}(\boldsymbol{P}'') - {}^{a}\boldsymbol{v}(\boldsymbol{P}')\left(\frac{\partial \boldsymbol{H}}{\partial t} + \frac{\partial \boldsymbol{H}}{\partial \boldsymbol{q}}\frac{\partial \boldsymbol{q}}{\partial t} + \frac{\partial \boldsymbol{H}}{\partial \boldsymbol{\theta}}\frac{\partial \boldsymbol{\theta}}{\partial t}\right)\right)$$

Hybrid dynamics

$$\begin{pmatrix} {}^{u}\dot{\boldsymbol{v}}(\boldsymbol{\xi}) \\ {}^{a}\boldsymbol{\tau}(\boldsymbol{\xi}) \end{pmatrix} = \boldsymbol{\mathcal{G}}(\ {}^{a}\boldsymbol{q}(\boldsymbol{P}),\ {}^{a}\boldsymbol{v}(\boldsymbol{P}'),\ {}^{a}\dot{\boldsymbol{v}}(\boldsymbol{P}''),\ {}^{u}\boldsymbol{\tau}(\boldsymbol{P}), \boldsymbol{\theta}(\boldsymbol{\xi})\ )$$

Unactuated Kinematics

$${}^{u}\dot{\boldsymbol{q}}(\boldsymbol{\xi}) = \ {}^{u}\boldsymbol{H}\left({}^{u}\boldsymbol{q}(\boldsymbol{\xi}), \boldsymbol{\theta}(\boldsymbol{\xi})\right) {}^{u}\boldsymbol{v}(\boldsymbol{\xi})$$

Outputs (81)

$$\boldsymbol{y}(\boldsymbol{\xi}) = \boldsymbol{\mathcal{O}}\left(\boldsymbol{q}(\boldsymbol{P};\boldsymbol{\xi}), \dot{\boldsymbol{q}}(\boldsymbol{P}';\boldsymbol{\xi}), \boldsymbol{\theta}(\boldsymbol{\xi})\right)$$

Constraints

$$\boldsymbol{\mathcal{C}}(\boldsymbol{y}(\boldsymbol{\xi}), \boldsymbol{\theta}(\boldsymbol{\xi}), \boldsymbol{\tau}(\boldsymbol{\xi})) \leq \mathbf{0}$$

Hard actuated ICs & TCs Conditions

$${}^{a}\boldsymbol{q}(0) = \ {}^{a}\boldsymbol{P}^0 = \ {}^{a}\boldsymbol{q}_0$$

$${}^{a}\dot{\boldsymbol{q}}(0) = \ {}^{a}\boldsymbol{P}^0 = \ {}^{a}\dot{\boldsymbol{q}}_0$$

$${}^{a}\boldsymbol{q}\left(t_f\right) = \ {}^{a}\boldsymbol{P}^{n_{sp}} = \ {}^{a}\boldsymbol{q}_{t_f}$$

$${}^{a}\dot{\boldsymbol{q}}\left(t_f\right) = \ {}^{a}\boldsymbol{P}'^{n_{sp}-1} = \ {}^{a}\dot{\boldsymbol{q}}_{t_f}$$

Hard unactuated ICs & TCs Conditions

$${}^{u}\boldsymbol{q}(0;\boldsymbol{\xi}) = \ {}^{u}\boldsymbol{q}_0(\boldsymbol{\xi})$$

$${}^{u}\dot{\boldsymbol{q}}(0;\boldsymbol{\xi}) = \ {}^{u}\dot{\boldsymbol{q}}_0(\boldsymbol{\xi})$$

$${}^{u}\boldsymbol{q}\left(t_f;\boldsymbol{\xi}\right) = \ {}^{u}\boldsymbol{q}_{t_f}(\boldsymbol{\xi})$$

$${}^{u}\dot{\boldsymbol{q}}\left(t_f;\boldsymbol{\xi}\right) = \ {}^{u}\dot{\boldsymbol{q}}_{t_f}(\boldsymbol{\xi})$$

Equations (79)–(81) are reformulations of (72)–(74) using the appropriate uncertain dynamics as discussed in Section 2.8. The *known* quantities of Table 5.1 remain deterministic, however, due to the present of the system uncertainties, all *unknown* quantities become uncertain and are modeled using the gPC techniques reviewed in Section 4.2. To help clarify this point, Table 5.1 is reproduced in Table 5.2 with the uncertainties explicitly illustrated.

Table 5.2—Deterministic Knowns vs Uncertain Unknowns

Formulation	**Known ($\boldsymbol{P}$)**	**Unknown (ξ)**
Forward	$\boldsymbol{\tau}(\boldsymbol{P})$	$\boldsymbol{q}(\boldsymbol{\xi}), \dot{\boldsymbol{q}}(\boldsymbol{\xi}), \ddot{\boldsymbol{q}}(\boldsymbol{\xi}), \boldsymbol{v}(\boldsymbol{\xi}), \dot{\boldsymbol{v}}(\boldsymbol{\xi})$
Inverse	$\boldsymbol{q}(\boldsymbol{P}), \dot{\boldsymbol{q}}(\boldsymbol{P}'), \ddot{\boldsymbol{q}}(\boldsymbol{P}''), \boldsymbol{v}(\boldsymbol{P}'), \dot{\boldsymbol{v}}(\boldsymbol{P}'')$	$\boldsymbol{\tau}(\boldsymbol{\xi})$
Hybrid	${}^{a}\boldsymbol{q}(\boldsymbol{P})$, ${}^{a}\dot{\boldsymbol{q}}(\boldsymbol{P}')$, ${}^{a}\ddot{\boldsymbol{q}}(\boldsymbol{P}'')$, ${}^{a}\boldsymbol{v}(\boldsymbol{P}')$, ${}^{a}\dot{\boldsymbol{v}}(\boldsymbol{P}'')$, ${}^{u}\boldsymbol{\tau}(\boldsymbol{P})$	${}^{u}\boldsymbol{q}(\boldsymbol{\xi})$, ${}^{u}\dot{\boldsymbol{q}}(\boldsymbol{\xi})$, ${}^{u}\ddot{\boldsymbol{q}}(\boldsymbol{\xi})$, ${}^{u}\boldsymbol{v}(\boldsymbol{\xi})$, ${}^{u}\dot{\boldsymbol{v}}(\boldsymbol{\xi})$, ${}^{a}\boldsymbol{\tau}(\boldsymbol{\xi})$

The most interesting part of (79)–(81) comes in the definition of the objective function terms and constraints. These terms now have the ability to approach the design accounting for uncertainties by way of expected values, variances, and standard deviations.

Recalling the definitions of an expected value and variance, (76)–(78) may be redefined statistically:

$$\mathrm{J_{S1}} = \sum_{i=1}^{n_i} \int_{t_0=0}^{t_f} E\left[\mathrm{z}_i\left(\tau_i(\boldsymbol{\xi},t)\right)^2\right] dt = \sum_{i=1}^{n_i} \int_{t_0=0}^{t_f} \sum_{j=0}^{n_b} \mathrm{w}_i\left(\tau_i^j(t)\right)^2 \langle \Psi^j, \Psi^j \rangle \, dt \tag{82}$$

$$\mathrm{J_{S2}} = \sum_{i=1}^{n_i} \int_{t_0=0}^{t_f} E[\,|\mathrm{z}_i\tau_i(\boldsymbol{\xi},t)\mathrm{y}_i(\boldsymbol{\xi},t)|]dt \; = \sum_{i=1}^{n_i} \int_{t_0=0}^{t_f} \sum_{j=0}^{n_b} \left|\mathrm{w}_i\tau_i^j(t)\mathrm{y}_i^j(t)\langle \Psi^{\mathrm{j}}, \Psi^j \rangle\right| dt \tag{83}$$

$$\mathrm{J_{S3}} = \sum_{i=1}^{n_i} \int_{t_0=0}^{t_f} E\left[\mathrm{z}_i\left(\dot{\tau}_i(\boldsymbol{\xi},t)\right)^2\right] dt \; = \sum_{i=1}^{n_i} \int_{t_0=0}^{t_f} \sum_{j=0}^{n_b} \mathrm{w}_i\left(\dot{\tau}_i^j(t)\right)^2 \langle \Psi^{\mathrm{j}}, \Psi^{\mathrm{j}} \rangle \, dt \tag{84}$$

where $\mathbf{w}$ is a vector of (optional) scalarization weights. The function (82) encapsulates the expected effort, (83) the expected power with $\mathrm{y}_i(\boldsymbol{\xi},t) = \dot{q}_i(\boldsymbol{\xi},t)$, and (84) the expected jerk. Close inspection of Table 5.2 shows that these statistically based objective function terms are applicable to the inverse and hybrid dynamics based motion planning formulations, (79)–(80).

Designs may necessitate statistically penalizing terminal conditions (TC) of the state or output trajectories in the objective function (occasionally referred to as *soft* constraints). Two candidates are,

$$\mathrm{J_{S4}} = \left\|\mu_{e(t_f)}\right\| = \left\|E\left[\boldsymbol{e}\left(t_f;\boldsymbol{\xi}\right)\right]\right\| \quad = \left\|\boldsymbol{y}_{ref}\left(t_f\right) - \boldsymbol{y}^0(t_f)\langle \Psi^0, \Psi^0 \rangle\right\| \tag{85}$$

$$\mathrm{J_{S5}} = \left\|\sigma_{e(t_f)}^2\right\| = \left\|E\left[\left(\boldsymbol{e}(\mathrm{t_f};\boldsymbol{\xi}) - \mu_{e(t_f)}\right)^2\right]\right\| = \left\|\sum_{j=0}^{n_b}\left(\boldsymbol{y}^j(\mathrm{t_f})\right)^2 \langle \Psi^j, \Psi^j \rangle\right\| \tag{86}$$

where $\boldsymbol{e}(\mathrm{t_f};\boldsymbol{\xi}) = \boldsymbol{y}_{ref}(\mathrm{t_f}) - \boldsymbol{y}(\mathrm{t_f};\boldsymbol{\xi})$; (85) is the expected value of the TC's error; (86) is the corresponding variance of the TC's error.

Due to the orthogonality of the polynomial basis, equations (82)–(86) result in a reduced set of efficient operations on their respective gPC expansion coefficients.

The inequality constraints may also benefit from added statistical information; for example, bounding the expected values can be expressed as,

$$\boldsymbol{C}(t;\boldsymbol{\xi}) = \underline{\boldsymbol{y}} \leq E[\boldsymbol{y}(\boldsymbol{\xi})] \leq \overline{\boldsymbol{y}} \tag{87}$$

where $E[\boldsymbol{y}(\boldsymbol{\xi})] = \mu_{\boldsymbol{y}} = \boldsymbol{y}^0\langle\Psi^j,\Psi^j\rangle$, and $\left\{\underline{\boldsymbol{y}},\overline{\boldsymbol{y}}\right\}$ are the minimum/maximum output bounds, respectively.

Collision avoidance constraints would ideally involve *supremum* and *infimum* bounds,

$$\underline{\boldsymbol{y}} \leq \inf\big(\boldsymbol{y}(t;\boldsymbol{\xi})\big),\ \ \sup(\boldsymbol{y}(t;\boldsymbol{\xi}))\ \leq \overline{\boldsymbol{y}} \tag{88}$$

However, one major difficulty with *supremum* and *infimum* bounds is that they are expensive to calculate. A more efficient alternative can be to constrain the uncertain configuration in a standard deviation sense; collision constraints would then take the form,

$$\begin{aligned} \mu_{\boldsymbol{y}} + \sigma_{\boldsymbol{y}}\ &\leq \overline{\boldsymbol{y}} \\ \underline{\boldsymbol{y}} &\leq \mu_{\boldsymbol{y}} - \sigma_{\boldsymbol{y}} \end{aligned} \tag{89}$$

where $std[\boldsymbol{y}(\boldsymbol{\xi})] = \sigma_{\boldsymbol{y}} = \sqrt{\sum_{j=1}^{n_b} \mathbf{y}^{\mathrm{j}}\langle\Psi^j,\Psi^j\rangle}$.

Therefore, the application of the appropriate equations from (79)–(89) enables a designer to treat all possible realizations of a given uncertain system when planning motion of fully-actuated and under-actuated systems.

5.5 Illustrating Case-Studies

This section presents case-studies which illustrate the benefits of the new motion planning framework for uncertain fully-actuated and under-actuated systems. Treatment of uncertainties during the motion planning phase allows designers to determine answers to new questions that previously were not possible, or very difficult, to answer. Three case-studies are presented; the first two are based on a fully-actuated serial manipulator 'pick-and-place' application (shown in Figure 5.1); the first of these uses the *forward dynamics* formulation (79); the second uses the *inverse dynamics* formulation (80). The third case-study illustrates the *hybrid dynamics* formulation (81) through an under-actuated inverting double pendulum problem (shown in Figure 5.8).

5.5.1 *Forward Dynamics* Based Uncertain Motion Planning

As an illustration of (79), the serial manipulator "pick-and-place" problem will be used (see Figure 5.1). The design objective is to minimize the effort it takes to move the manipulator from its initial configuration, $\boldsymbol{q}_0$, to the target configuration, $\boldsymbol{q}_{t_f}$ in a prescribed amount of time, t_f. This results in a deterministic objective function of, $J = \sum_{i=1}^{n_i} \mathrm{z}_i \tau^2$, which is frequently referred to as an *effort optimal* design. However, the payload mass, $\mathrm{M}(\xi)$, is defined to be uncertain rendering the system dynamics uncertain. Since the uncertain serial manipulator is a fully actuated system, where the joints $\boldsymbol{q} = \{q_1, q_2\}$ are actuated with the input wrenches $\boldsymbol{\tau} = \{\tau_1, \tau_2\}$, the motion planning problem may be appropriately defined by (79).

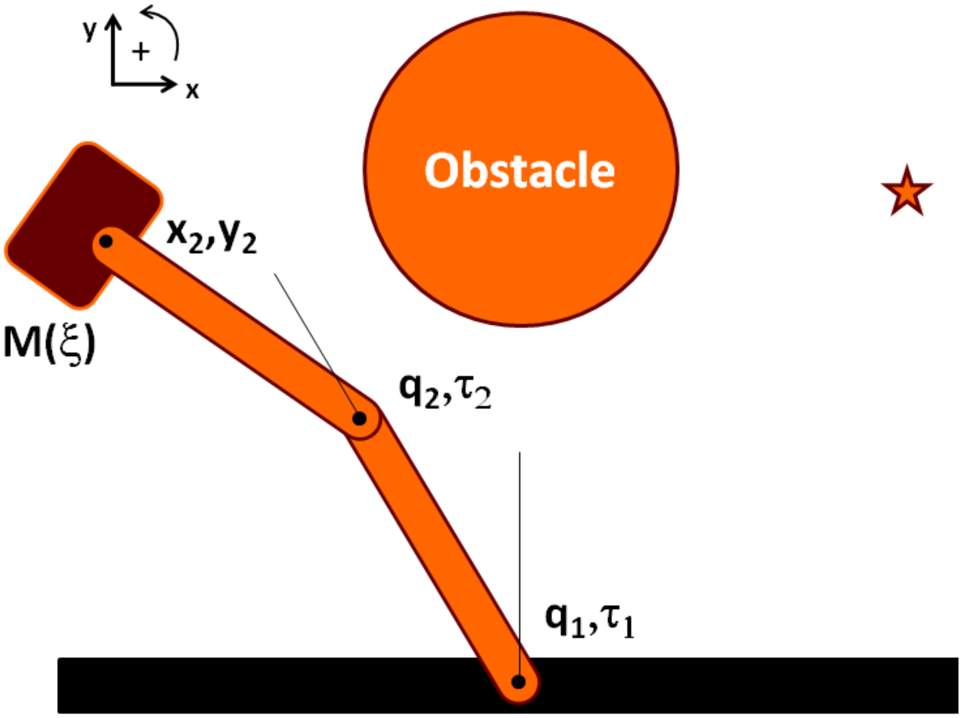

Figure 5.1—A simple illustration of an uncertain fully-actuated motion planning problem; the *forward dynamics* based formulation aims to determine an *effort optimal* motion plan; the *inverse dynamics* based formulation aims to determine a *time optimal* motion plan. Both problems are subject to input wrench and geometric collision constraints. This system is an uncertain system due to the uncertain mass of the payload.

By parameterizing the input wrench profiles with B-Splines, in a similar fashion as (59), (79) results in a finite search problem seeking for spline control points, $\mathbf{P}$, that minimize the actuation effort defined in J. Therefore, the problem's optimization variables are $\boldsymbol{x} = \{\mathbf{P}\}$.

The actuators are bounded in their torque supply and the manipulator should neither hit the wall it's mounted to nor the obstacle. The constraints may therefore be defined as,

$$\boldsymbol{C}: \begin{cases} \underline{\boldsymbol{\tau}} \leq \boldsymbol{\tau} \leq \bar{\boldsymbol{\tau}} \\ \mu_{\boldsymbol{y}} \pm \sigma_{\boldsymbol{y}} \leq \mathbf{0} \\ -\mathfrak{D}_{i,j}(\mu_{\boldsymbol{y}} \pm \sigma_{\boldsymbol{y}}) \leq 0 \end{cases} \tag{90}$$

where $i = 1{,}2$ and $j = obstacle$ for the signed distance, $\mathfrak{D}_{i,j}(\mu_{\boldsymbol{y}} \pm \sigma_{\boldsymbol{y}})$, measured from each link of the serial manipulator to the obstacle calculated using the statistical mean and standard deviations of the configuration/outputs; and $\{\underline{\tau}, \bar{\tau}\}$ are the minimum/maximum input bounds, respectively.

This formulation allows a design engineer to answer the question,

> *Given actuator and obstacle constraints, what is the "effort optimal" motion plan that accounts for all possible systems within the probability space?*

Without accounting for the uncertainty directly in the dynamics and motion planning formulations, design engineers would have a difficult time answering this question. As a result, manufacturing lines, or other applicable applications, would result in reduced yield rates potentially affecting the company's financial *bottom-line*.

The solution to this problem with the deterministic formulation, as defined in (72), results in an *effort optimal* solution of $J = 2770\ (\mathrm{Nm})^2$; where $\mathrm{t_f} = 1.5\ \mathrm{seconds}$; all system parameters are set equal to one, $\theta_{\mathrm{i}} = 1$ (with SI units); initial conditions $\boldsymbol{q}(0) = \{\frac{\pi}{6}, \frac{\pi}{6}\}$ (rad) and $\dot{\boldsymbol{q}}(0) = \{0, 0\}$ (rad/sec); terminal conditions $\boldsymbol{q}(\mathrm{t_f}) = \{-\frac{\pi}{6}, -\frac{\pi}{6}\}$ (rad) and $\dot{\boldsymbol{q}}(\mathrm{t_f}) = \{0, 0\}$ (rad/sec); and $\underline{\tau} = -10, \bar{\tau} = 10$ (Nm). The resulting optimal configuration time history is shown in Figure 5.2.

The solution from the new formulation, as defined in (79) with constraints defined by (90), results in an *effort optimal* solution of $J = 3530\ (\mathrm{Nm})^2$; where all system parameters and initial/terminal conditions are defined the same as in the deterministic problem. The only difference in this problem definition, as compared to the deterministic problem, is the uncertain pay-load mass modeled with a uniform distribution having a unity mean and 0.5 variance. The uniform distribution for the uncertain payload and corresponding dynamic states and outputs were approximated with a fifth order, $p_o = 5$, Legrendre polynomial serias expansion. This corresponds to six basis terms, $n_b = 6$, where $3n_b$ collocation points were generated using a Hammersley *low-discrepancy sequence* (LDS) before being solved by the LSCM. The resulting optimal uncertain end-effector Cartesian position time history is illustrated in Figure 5.3; where the mean and bounding $\mu_y \pm \sigma_y$ time histories are displayed.

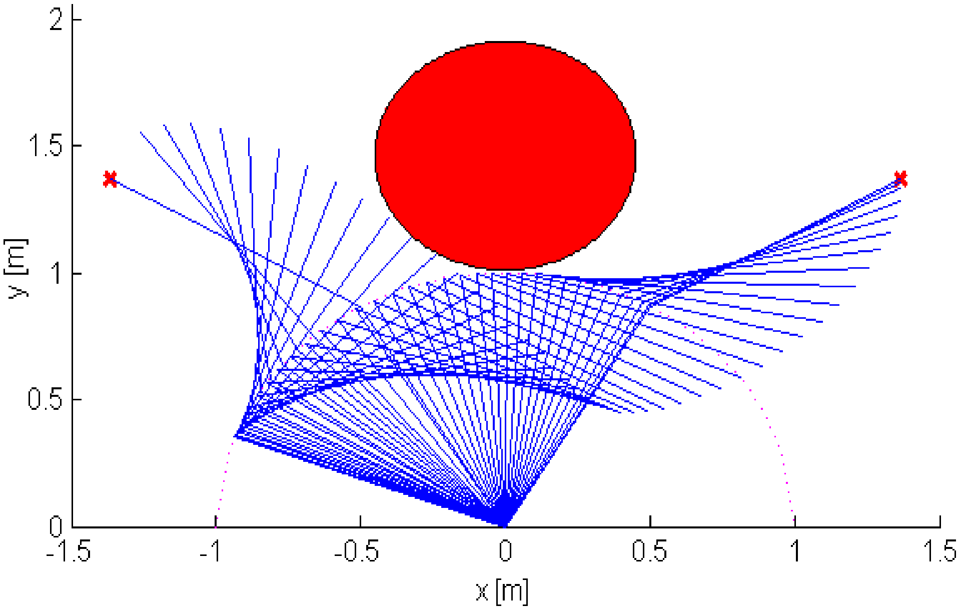

Figure 5.2—The *effort optimal* configuration time histories for the deterministic serial manipulator 'pick-and-place' problem. This optimal solution resulted in a $J = 2770$ (Nm)^2 design.

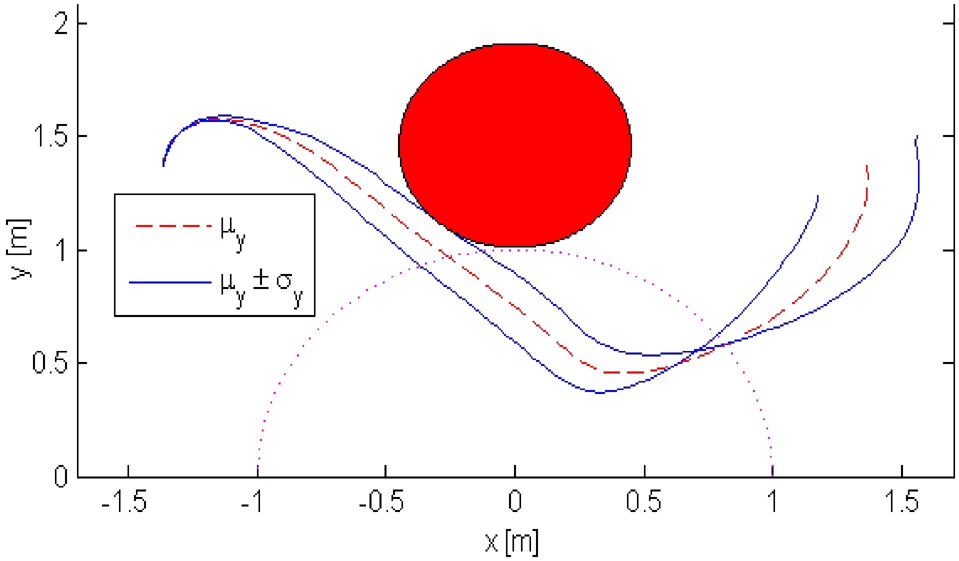

Figure 5.3—The *effort optimal* uncertain end-effector Cartesian position time history for the uncertain serial manipulator 'pick-and-place' problem based on the *uncertain forward dynamics* NLP. The mean and bounding $\mu_y \pm \sigma_y$ time histories are displayed. This optimal solution resulted in a $J = 3530\ (\mathrm{Nm})^2$ design.

Therefore, the *effort optimal* solution from the uncertain problem resulted in a more conservative answer—$3530\ (\mathrm{Nm})^2$ as compared to $2770\ (\mathrm{Nm})^2$. This is a sensible solution; close inspection of Figure 5.2 shows the deterministic solution drove the configuration as close to the obstacle as possible. The introduction of uncertainty in the pay-load mass affected the amount of input torque required for the system to reliably avoid the obstacle for all systems within the probability space. In fact, Figure 5.3 shows the distribution of end-effector Cartesian position trajectory induced by the uncertain pay-load. The uncertain optimal motion plan from (79) effectively pushed the end-effector configuration distribution away from the obstacle; this results in a larger *effort optimal* solution, however, all realizable systems within the probability space of the uncertain mass are now guaranteed to satisfy the constraints. In other words, the *effort optimal* solution to (79) produces the minimum effort design for the entire family of systems. Relying only on the contemporary deterministic problem formulation in (72) results in an unrealizable trajectory for a subset of the realizable systems.

A third study provides some additional insight to what the new framework can provide. By redefining the objective function for (79) as (86) the uncertain design is no longer an *effort optimal* but *terminal variance optimal* design. In other words, the new design question is,

> *Given actuator and obstacle constraints, what motion plan will minimize the variance of the terminal condition's (TC) error when accounting for all possible systems within the probability space?*

The *effort optimal* design resulted in a TC error standard deviation of $\sigma_{e(t_f)} = [0.191, 0.133]\ (m)$; where the standard deviation is the square root of the variance. Redesigning the motion plan using an objective function defined by (86) results in a TC error standard deviation of $\sigma_{e(t_f)} = [0.144, 0.114]\ (m)$, as shown in Figure 5.4. Therefore, a modest reduction in the TC error standard deviation was realized, however, the effort of the new design increased from $3530\ (\mathrm{Nm})^2$ to $5910\ (\mathrm{Nm})^2$. These results indicate a Pareto optimal trade-off between the effort and TC's variance. Therefore, designers may define a hybrid objective function with a scalarization between the *effort optimal* and *terminal variance optimal* terms.

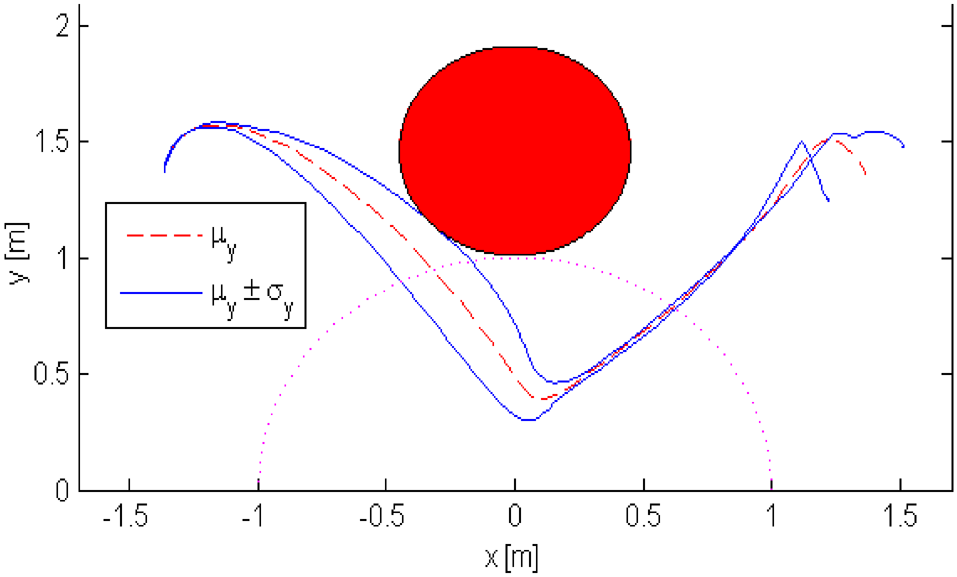

Figure 5.4—The *terminal variance optimal* uncertain end-effector Cartesian position time history for the uncertain serial manipulator 'pick-and-place' problem based on the *uncertain forward dynamics* NLP. The mean and bounding $\mu_y \pm \sigma_y$ time histories are displayed. This optimal solution resulted in a $J = 5910\ (\mathrm{Nm})^2$ design.

One additional insight gained from the *terminal variance optimal* design is related to the controllability of an uncertain system's TC variance. If the TC variance was fully controllable then the *terminal variance optimal* design would be able to reduce it to zero. This initial investigation indicates that the variance is not fully controllable. A rigorous uncertain system controllability investigation is out of the scope of this work but will be considered for future research.

A final observation is that the *uncertain forward dynamics* motion planning framework embodied in (79) is most applicable to force controlled systems where input wrenches are prescribed. However, configuration/position controlled systems may be better designed through application of the *uncertain inverse dynamics* based NLP found in (80); this is illustrated in the next section.

5.5.2 *Inverse Dynamics* Based Uncertain Motion Planning

As an illustration of (80), the serial manipulator "pick-and-place" problem is re-used (see Figure 5.1). The design objective is to minimize the time it takes to move the manipulator from its initial configuration, $\boldsymbol{q}_0$, to the target configuration, $\boldsymbol{q}_{t_f}$. This results in a deterministic objective function, $J = \mathrm{t_f}$, which is frequently referred to as a *time optimal* design. However, the payload mass, $\mathrm{M}(\xi)$, is defined to be uncertain rendering the system dynamics uncertain. Since the uncertain serial manipulator is a fully actuated system, where the joints $\boldsymbol{q} = \{q_1, q_2\}$ are actuated with the input wrenches $\boldsymbol{\tau} = \{\tau_1, \tau_2\}$, the motion planning problem may be appropriately defined by (80).

By parameterizing the deterministic joint trajectories with B-Splines, as in (59), (80) results in a finite search problem seeking for spline control points, $\mathbf{P}$, that minimize the trajectory time, $\mathrm{t_f}$. Therefore, the problem's optimization variables are $\boldsymbol{x} = \{\mathbf{P}, \mathrm{t_f}\}$.

The actuators are bounded in their torque supply and the manipulator should neither hit the wall it's mounted to nor the obstacle. The constraints may therefore be defined as,

$$\boldsymbol{\mathcal{C}}: \begin{cases} \mu_{\tau_i} + \sigma_{\tau_i} \leq \bar{\tau} \\ \underline{\tau} \leq \mu_{\tau_i} - \sigma_{\tau_i} \\ -\mathrm{y}_1 \leq 0 \\ -\mathrm{y}_2 \leq 0 \\ -\mathfrak{D}_{i,j} \leq 0 \end{cases} \tag{91}$$

where $i = 1{,}2$ and $j = obstacle$ for the signed distance, $\mathfrak{D}_{i,j}$, measured from each link of the serial manipulator to the obstacle.

Notice the bounding constraints on the input wrenches are defined by their statistical mean and standard deviations, as in (89), to quantify their uncertainty. Ideally these constraints would be defined by the extremes of the wrench distribution (i.e., the *supremum* and the *infimum*), however, due to their computational complexity the approximation by the mean and standard deviation, as in (91), is used.

Since the state trajectories are deterministic, the signed obstacle avoidance constraints, $-\mathfrak{D}_{i,j} \leq 0$, and Cartesian wall avoiding constraints, $-\mathrm{y}_1, -\mathrm{y}_2 \leq 0$, are deterministically defined.

This formulation allows a design engineer to answer the question,

> *Given actuator and obstacle constraints, what is the "time optimal" motion plan that accounts for all possible systems within the probability space?*

Without accounting for the uncertainty directly in the dynamics and motion planning formulations, design engineers would have a difficult time answering this question.

The solution to this problem with the deterministic formulation, as defined in (73), results in a *time optimal* solution of $\mathrm{t_f} = 1.12$ seconds; where all system parameters are set equal to one, $\theta_\mathrm{i} = 1$ (with SI units); with initial conditions $\boldsymbol{q}(0) = \{\frac{\pi}{6}, \frac{\pi}{6}\}$ (rad) and $\dot{\boldsymbol{q}}(0) = \{0, 0\}$ (rad/sec); terminal conditions $\boldsymbol{q}(\mathrm{t_f}) = \{-\frac{\pi}{6}, -\frac{\pi}{6}\}$ (rad) and $\dot{\boldsymbol{q}}(\mathrm{t_f}) = \{0, 0\}$ (rad/sec); and $\underline{\tau} = -10, \bar{\tau} = 10$ (Nm). The resulting optimal input wrench time history is shown in Figure 5.5.

The solution from the new formulation, as defined in (80) with constraints defined by (80), results in a *time optimal* solution of $\mathrm{t_f} = 1.2$ seconds; where all system parameters and initial/terminal conditions are defined the same as in the deterministic problem. The only difference in this problem definition, as compared to the deterministic problem, is the uncertain payload mass is modeled with a uniform distribution having a 1 (kg) mean and 0.5 (kg) standard deviation. As in Section 5.5.1, the uniform distribution for the uncertain payload and corresponding input wrench were approximated with a fifth order, $p_o = 5$, Legrendre polynomial serias expansion. This corresponds to six basis terms, $n_b = 6$, where $3n_b$ collocation points were generated using a Hammersley *low-discrepancy sequence* (LDS) before being solved by the LSCM. The resulting optimal uncertain input wrench time history is illustrated in Figure 5.6; where each input wrench is displaying its mean value and bounding $\mu_{\tau_i} \pm \sigma_{\tau_i}$ time histories. Also, the resulting configuration time history for the optimal uncertain motion plan is shown in Figure 5.7.

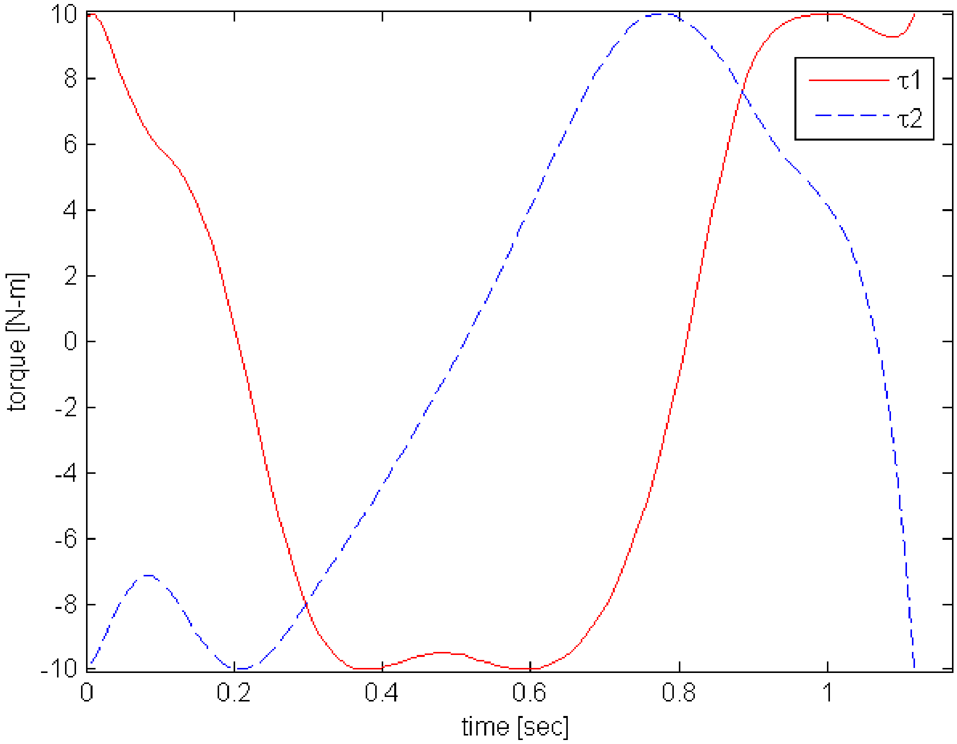

Figure 5.5—The *time optimal* input wrench time histories for the deterministic serial manipulator 'pick-and-place' problem based on the *uncertain inverse dynamics* NLP. This optimal solution resulted in a $t_f = 1.12$ (s).

Therefore, the *time optimal* solution from the uncertain problem resulted in a more conservative answer (1.2 seconds as compared to 1.12 seconds). This is a sensible solution; close inspection of Figure 5.5 shows the deterministic solution drove the input wrenches to their extreme bounds of +/-10 (Nm) at certain points during the motion profile. Clearly, introducing the uncertain mass to the system affected the amount of input torque required for the system to reliably follow the specified state trajectory. In fact, Figure 5.6 shows the distribution of input wrenches induced by the uncertain mass. The uncertain optimal motion plan from (80) effectively pushed the input wrench distribution inside the actuation limits, $\{\underline{\tau}, \bar{\tau}\}$; this results in a slower *time optimal* solution, however, the family of systems within the probability space of the uncertain mass are now guaranteed to satisfy the constraints (within an averaged standard deviation sense). In other words, the *time optimal* solution to (80) produces the minimum time motion plan that can be feasibly realized by the entire family of systems (in an averaged standard deviation sense). Relying only on the contemporary deterministic problem formulation in (73) results in an unrealizable trajectory for a subset of the realizable systems.

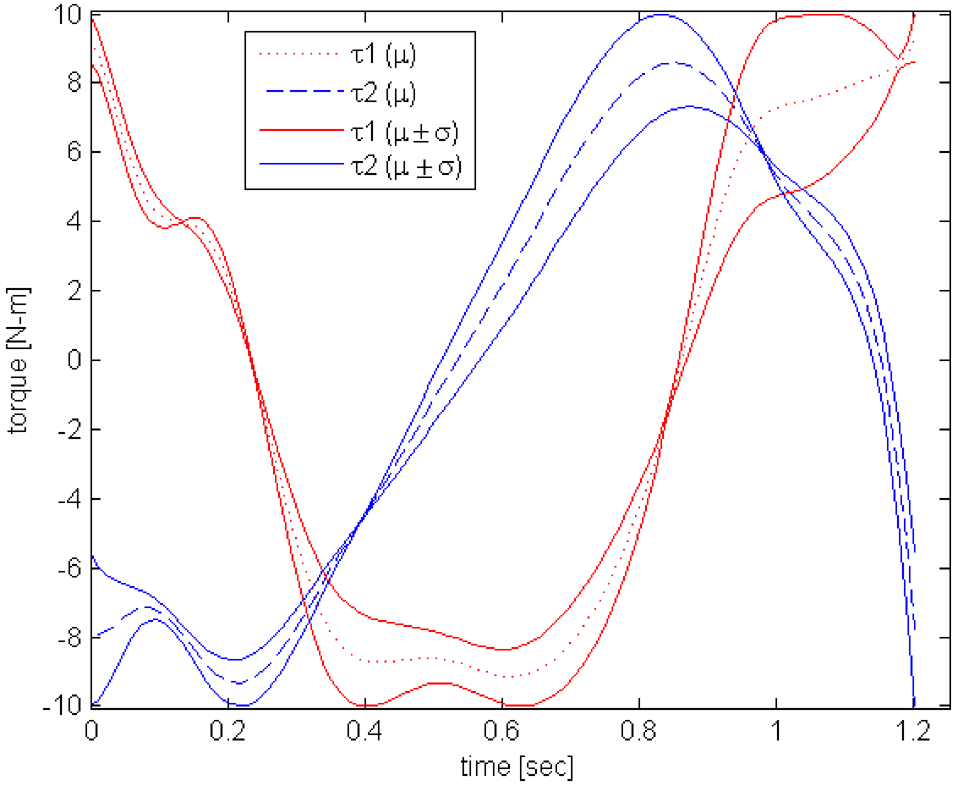

Figure 5.6—The time optimal uncertain input wrench time histories for the uncertain serial manipulator 'pick-and-place' problem based on the *uncertain inverse dynamics* NLP. Each input wrench is displaying its mean value and bounding $\mu_{\tau_i} \pm \sigma_{\tau_i}$ time histories. This optimal solution resulted in a $\mathrm{t_f} = 1.2$ (s).

A final observation is that the *uncertain inverse dynamics* motion planning framework embodied in (80) is most applicable to configuration/position controlled systems, where states are prescribed as they are in (59). However, force controlled systems may be better designed through application of (79) based on *uncertain forward dynamics* as illustrated in the previous section, Section 5.5.1.

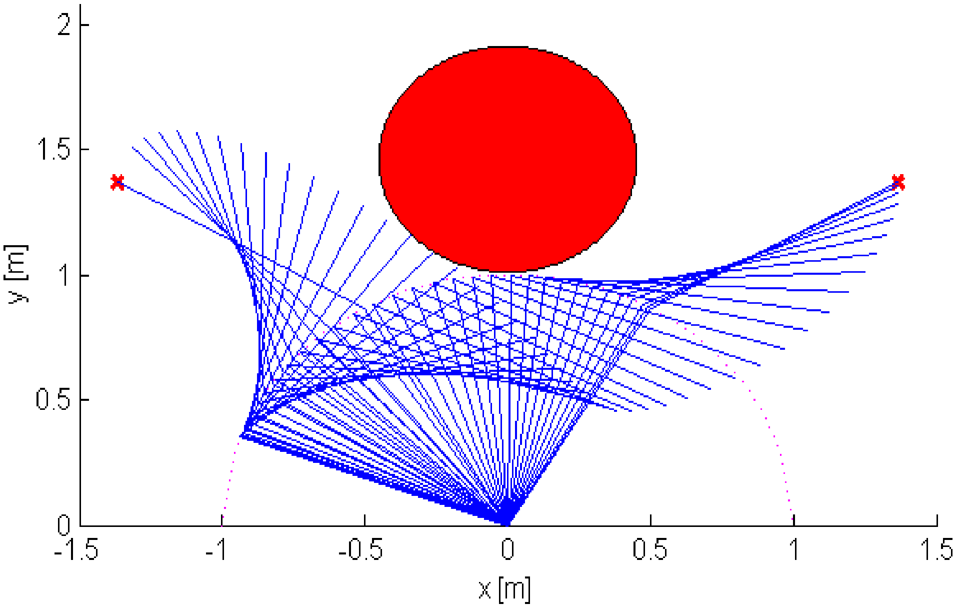

Figure 5.7—The final optimal configuration time history of the uncertain serial manipulator 'pick-and-place' application involving collision avoidance and actuator constraints design with the *uncertain inverse dynamics* NLP.

5.5.3 Hybrid Dynamics Based Uncertain Motion Planning

As an illustration of (81), an inverting double pendulum problem will be used (see Figure 5.8). The design objective is to minimize the power it takes to move the manipulator from its initial hanging configuration, $\boldsymbol{q}_0$, to the target inverted configuration, $\boldsymbol{q}_{t_f}$. The double pendulum is an under-actuated system, where only joint q_1 is actuated (by input wrench τ_1), and the mass of the second link is uncertain—modeled with a uniformly distributed PDF—therefore, the motion planning problem may be appropriately defined by (81).

By parameterizing the actuated state profiles with B-Splines, as in (59), and using the *hybrid dynamics* defined in (71), (81) results in a finite search problem seeking for spline control points, $\mathbf{P}$, and terminal time, t_f, that minimize the system's power. Therefore, the problem's optimization variables are $\boldsymbol{x} = \{\mathbf{P}, t_f\}$. Assuming a *soft* terminal error expected value condition is used, the objective function becomes $J = a \cdot \mathrm{J}_{S2} + b \cdot \mathrm{J}_{S4}$ from (83)–(85); where *a* and *b* are scalarization constants.

The actuators are bounded in their torque supply. Additionally, suppose the design has a specified variance in the terminal error conditions (86) that must be satisfied. Implementing both of these design constraints as *hard* constraints takes the form,

$$\boldsymbol{C}: \begin{cases} \underline{\boldsymbol{\tau}} \leq \boldsymbol{\tau} \leq \overline{\boldsymbol{\tau}} \\ \sigma^2_{\boldsymbol{e}(t_f)} \leq \overline{\sigma}^2_{\boldsymbol{e}(t_f)} \end{cases} \tag{92}$$

where $\{\underline{\tau}, \overline{\tau}\}$ are the minimum/maximum input bounds respectively; $\overline{\sigma}^2_{\boldsymbol{e}(t_f)}$ is the maximum terminal error variance.

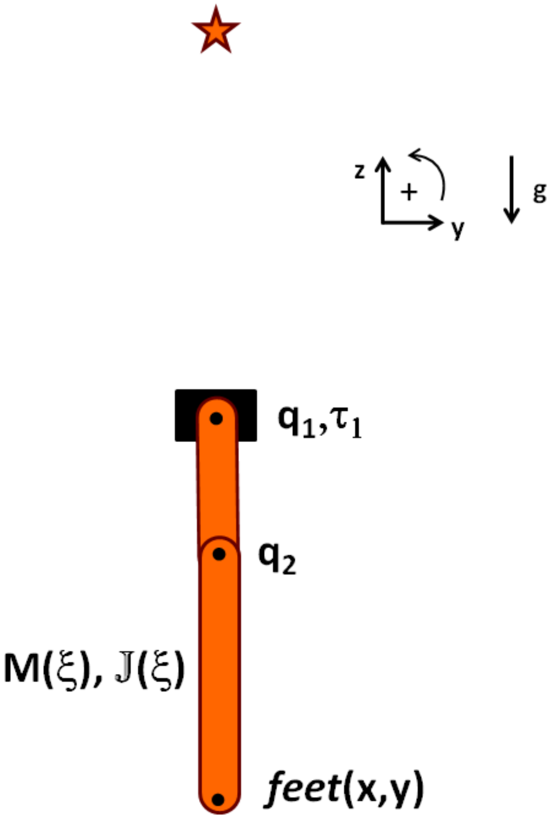

Figure 5.8—A simple illustration of the under-actuated *uncertain hybrid dynamics* motion planning formulation; this problem aims to determine a *power optimal* motion plan subject to input wrench and terminal condition constraints. This is an uncertain system due to the uncertain mass of the payload.

This formulation allows a design engineer to answer the question,

> *Given actuator and terminal error variance constraints, what motion plan will minimize the system's power over the trajectory when accounting for all possible systems within the probability space?*

Without accounting for the uncertainty directly in the dynamics and motion planning formulations, design engineers would have a difficult time answering this question.

The solution to this problem with the deterministic formulation, as defined in (74), results in an *power optimal* solution of $\mathrm{J_{S1}} = 1060\ (W)$ with $\mathrm{t_f} = 5.66$ seconds; all system parameters are set equal to one, $\theta_\mathrm{i} = 1$ (with SI units) except the length of the first link is set to 0.5 (m); initial conditions $\boldsymbol{q}(0) = \{-\pi, 0\}$ (rad) and $\dot{\boldsymbol{q}}(0) = \{0, 0\}$ (rad/sec); terminal conditions

$\boldsymbol{q}(\mathrm{t_f}) = \{0, 0\}$ (rad) and $\dot{\boldsymbol{q}}(\mathrm{t_f}) = \{0, 0\}$ (rad/sec); and the input limits are $\underline{\tau} = -10, \bar{\tau} = 10\ (N \cdot m)$. The resulting optimal motion plan's configuration time history is shown in Figure 5.9.

The value of the new framework is best illustrated by applying the deterministically designed motion profile to an uncertain system. Figure 5.10 and Figure 5.11 show the results of the deterministic motion plan applied to a system with a single uncertainty; the second link has a uniformly distributed uncertain mass with $\mu_{m2} = 1\ (kg)$ and $\sigma_{m2}^2 = 0.5\ (kg^2)$. Figure 5.10 shows that the resulting input wrench profile exceeds both the upper and lower bounding constraints of $\underline{\tau} = -10, \bar{\tau} = 10\ (N \cdot m)$. Additionally, Figure 5.11 shows that the target terminal configuration was not satisfied and an excessive terminal error variance is experienced.

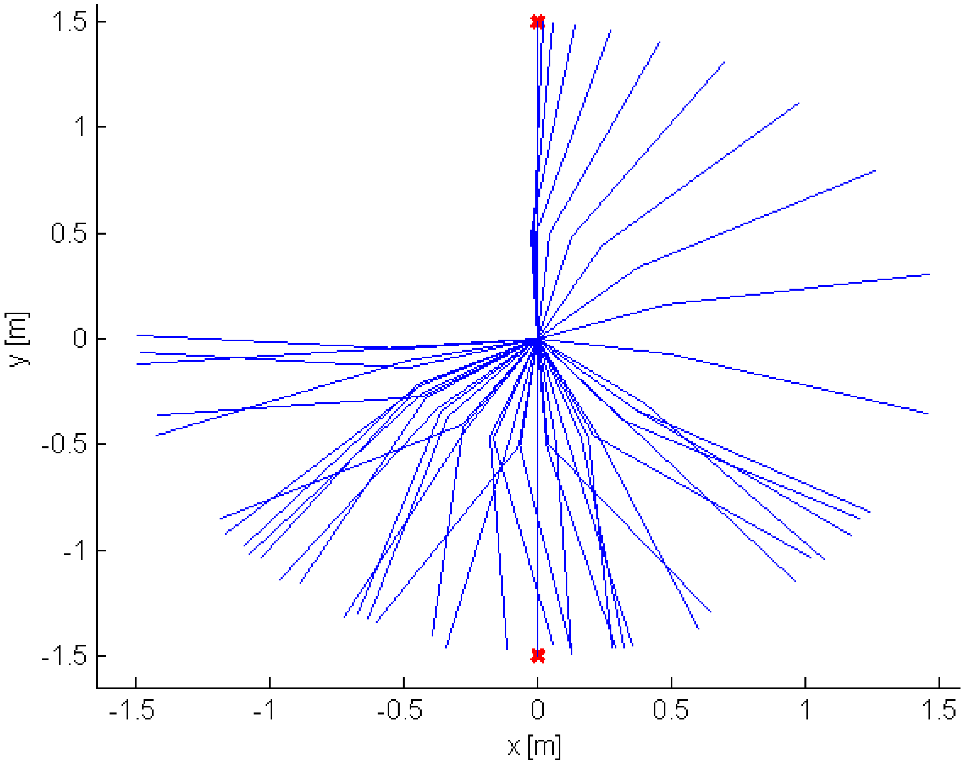

Figure 5.9—The *power optimal* configuration time history for the deterministic inverting double pendulum. This optimal solution resulted in a $1060\ (W)$ design.

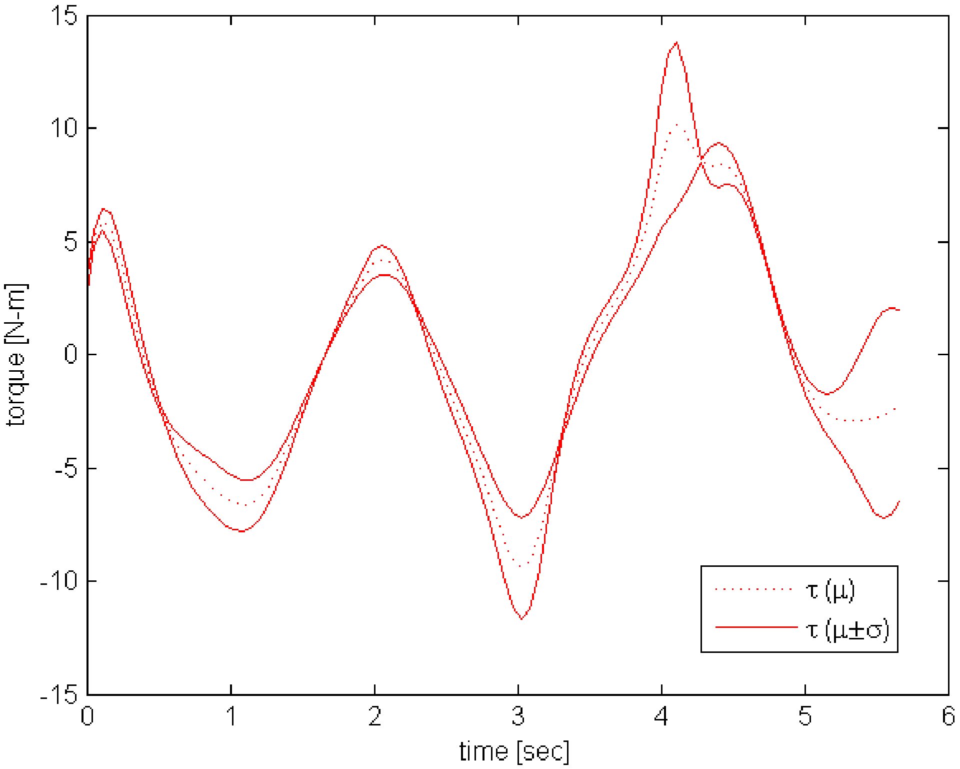

Figure 5.10—The uncertain input wrench time history for the deterministically design motion plan applied to an uncertain inverting double pendulum. The presence of the uncertainty results in both the maximum and minimum input limits being exceeded.

Approaching the design with the new framework accounts for the uncertainties up front during the optimal search and results in a design that satisfies all constraints for all possible systems in the probability space. This is accomplished by application of (81) with constraints defined by (92); where $\overline{\sigma}^2_{\boldsymbol{e}(t_f)} = 0.01\ (m^2)$. The uniform distributions for the uncertain second link mass and corresponding dynamic states and input wrenches were approximated with a fifth order, $p_o = 5$, Legrendre polynomial serias expansion. This corresponds to six basis terms, $n_b = 6$, where $3n_b$ collocation points were generated using a Hammersley *low-discrepancy sequence* (LDS) before being solved by the LSCM. This results in a *power optimal* solution of $\mathrm{J}_{S2} = 310\ (W)$ with $t_f = 4.46$ seconds; where the same uncertain second link mass is reused. The resulting motion plan's optimal uncertain configuration time history is illustrated in Figure 5.12; where the bounding $\{\mu_y - \sigma_y\ (red),\ \mu_y + \sigma_y\ (blue)\}$ configuration time histories are displayed. The Euclidean norm of the *soft* expected value terminal configuration constraint was very acceptable, $\left\|E\left[\boldsymbol{e}(t_f)\right]\right\| = 2.61e-6\ (m)$. Figure 5.13 shows that the input wrench constraints for the entire probability space were satisfied in a standard deviation sense. Figure

5.14 show that the specified terminal error variance was also satisfied, $\sigma^2_{e(t_f)} = 0.00321 \leq \overline{\sigma}^2_{e(t_f)} = 0.01\ (m^2)$.

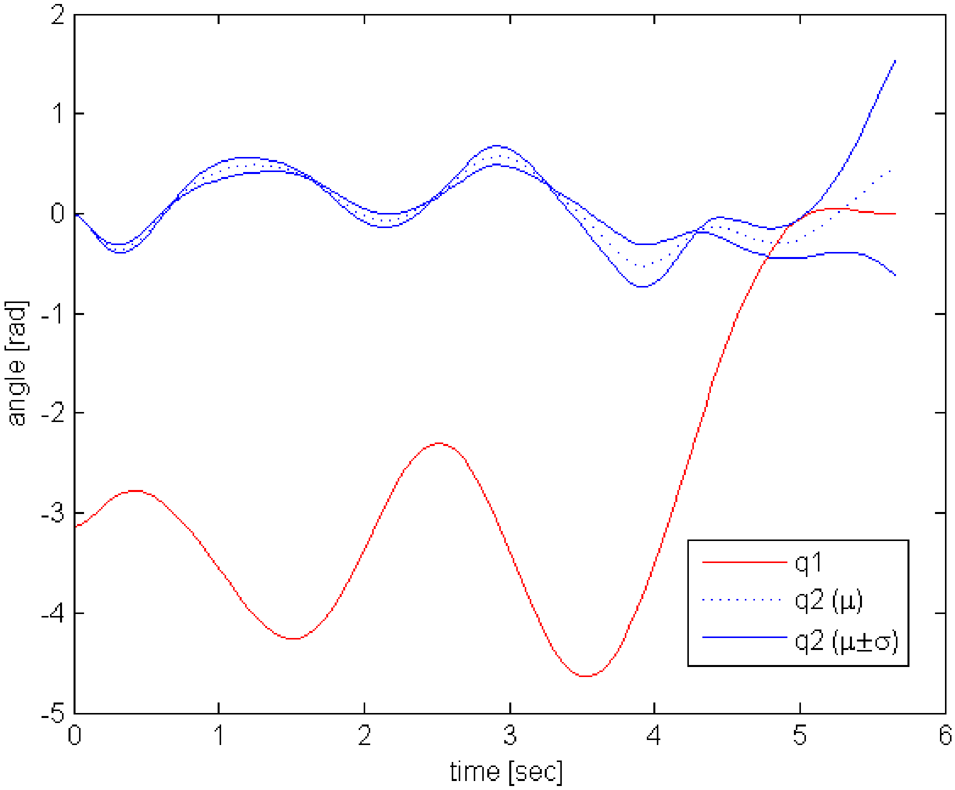

Figure 5.11—The joint time histories for the deterministically design motion plan applied to an uncertain inverting double pendulum. The presence of the uncertainty results in the expected terminal error condition not being satisfied with excessive variance.

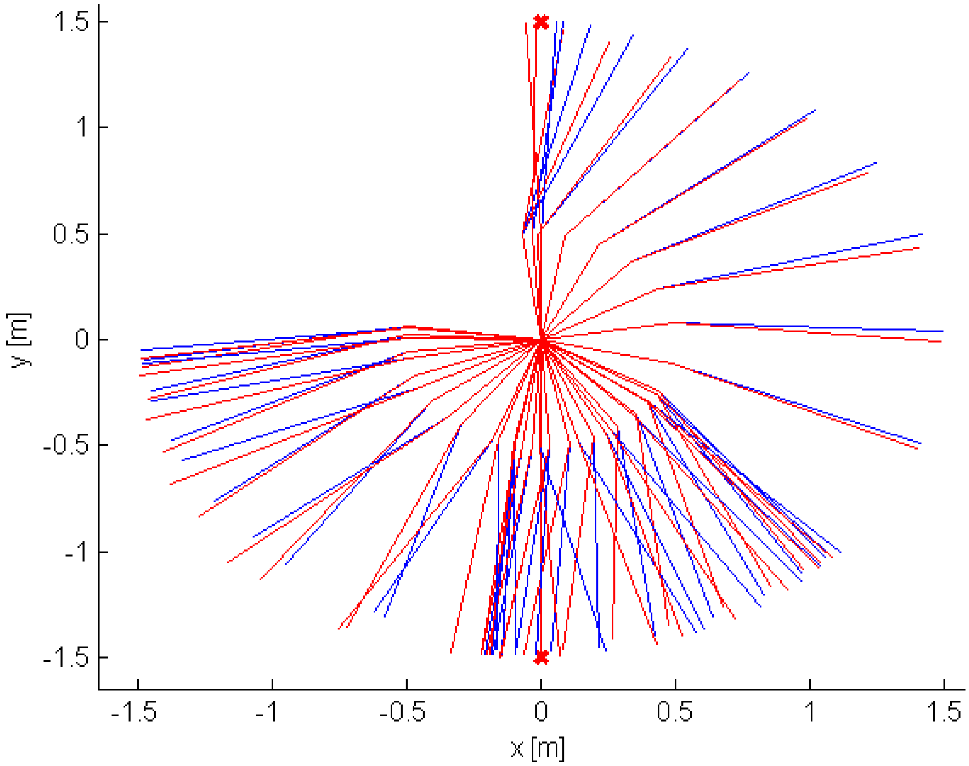

Figure 5.12—The *power optimal* configuration time history for the uncertain inverting double pendulum based on *uncertain hybrid dynamics* NLP. This optimal solution resulted in a 310 $(\boldsymbol{W})$ design.

The reduced power of the uncertain design, as compared to the deterministic design, makes sense in that the expected input wrench values, $E[\tau_1]$, of the uncertain design (as shown in Figure 5.13), are lower than those in the deterministic design (as shown in Figure 5.10). This relationship is also true for $\dot{q}_1$ (although are not illustrated), therefore, the product of the reduced expected torque and joint rate yields a lower system power.

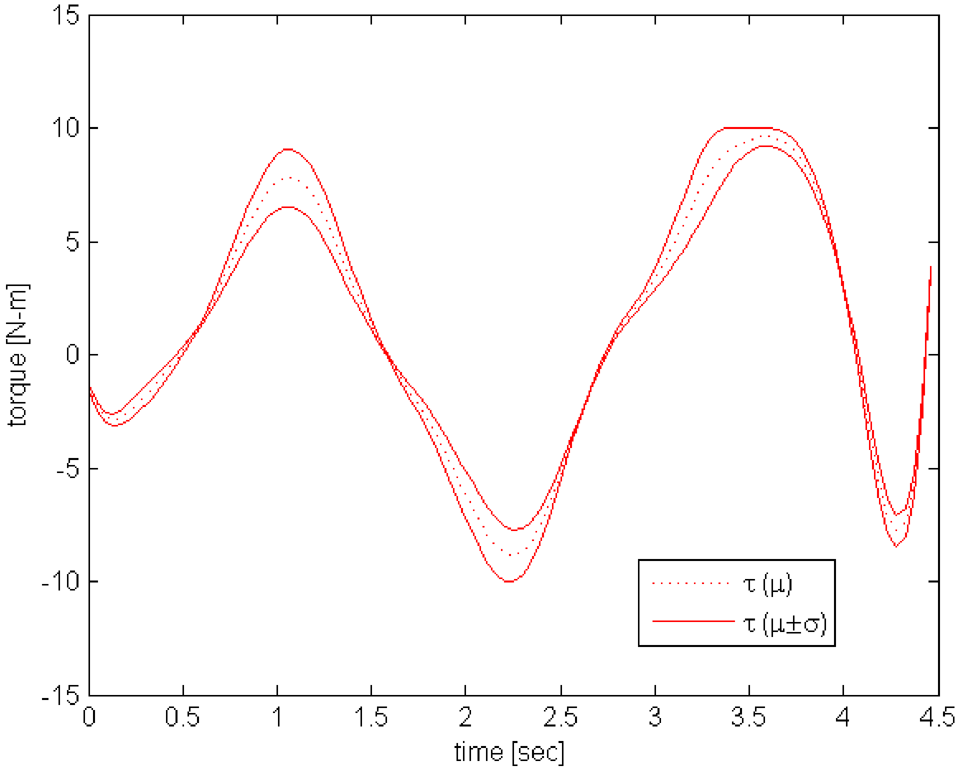

Figure 5.13—The uncertain input wrench time history resulting from the motion plan generated by the new *uncertain hybrid dynamics* NLP. Both the maximum and minimum input limits were satisfied, in a standard deviation sense, for all systems within the probability space.

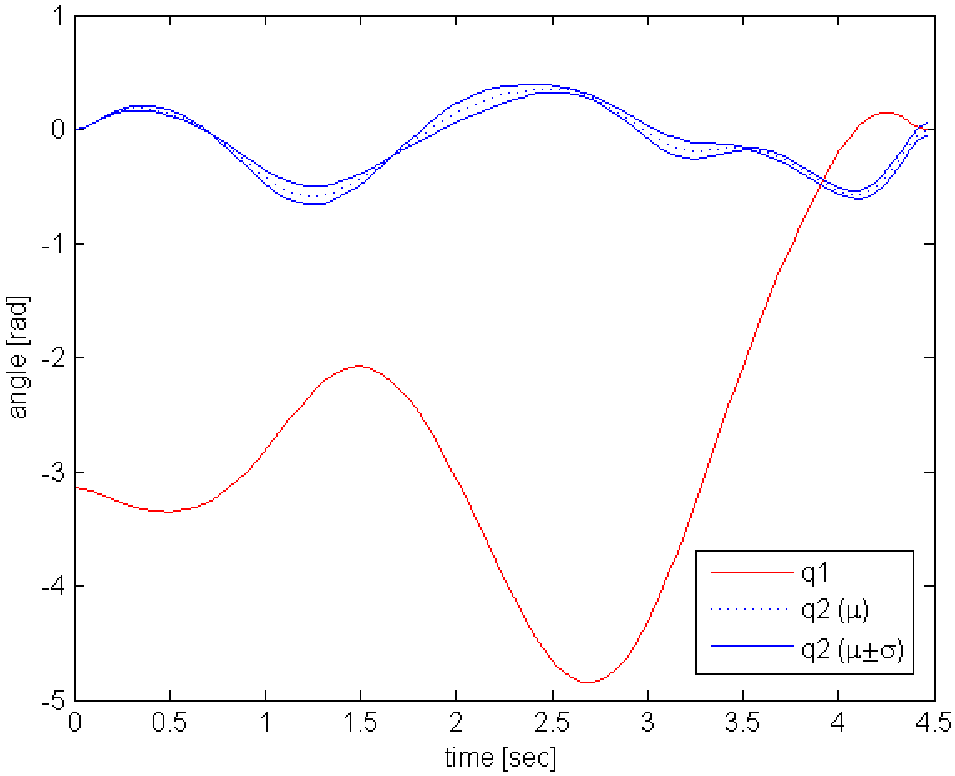

Figure 5.14—The joint time histories resulting from the motion plan generated by the new *uncertain hybrid dynamics* NLP. The resulting terminal error variance satisfies the specification; $\boldsymbol{\sigma^2_{e(t_f)} = 0.0032 \leq \overline{\sigma}^2_{e(t_f)} = 0.01\ (m^2)}$.

5.6 Uncertainty Skewness Considerations In Optimal Design

The results of the *forward*, *inverse*, and *hybrid dynamics* case studies all presented the time evolution of dependent uncertain quantities in a $(\mu \pm \sigma)$ format. It is important to mention that as a PDF is propagated through a nonlinear function it cannot, in general, be assumed that the skewness of the resulting PDF will remain consistent with that of the input PDF. Therefore, in an ideal optimal design setting, the corresponding skewness moments would be included in the uncertain constraint definitions (e.g., the collision avoidance and/or actuator torque limits). However, calculations of the skewness moment does not benefit from the orthogonality of the gPC approximation as the mean and variance calculations do; the skewness calculation is more computationally intensive. Therefore, the approach used in these investigations makes use of a constraint definition that does not account for the skewness. Once an optimal design has been determined, the skewness moments may be calculated for the resulting design and used to analyze, more accurately, the design's constraint satisfaction. If the constraint satisfaction is insufficient, or too conservative, then an appropriate scaling constant α may be used in the constraint definition, such as, $(\mu \pm \alpha\, \sigma)$, such that an iterated design may seek to improve the constraint satisfaction.

5.7 Conclusions

This section has presented a new NLP based framework for optimal motion planning that treats uncertain fully-actuated and under-actuated dynamical systems described by ODEs. The framework allows practitioners to model sources of uncertainty using the gPC methodology and to solve the uncertain forward, inverse, and hybrid dynamics using a LSCM. Subsequently, statistical moments from the uncertain dynamics may be included in the NLP’s objective function and constraints to perform optimal motion planning under uncertainty. Three case-studies with uncertain dynamics illustrate how the new framework produces an optimal design that accounts for the entire family of systems within the associated probability space. This adds robustness to the design of the optimally performing system.

6 Parametric Design Optimization of Uncertain ODE Systems

6.1 Motivation

Design engineers cannot quantify exactly every aspect of a given system. These uncertainties frequently create difficulties in accomplishing design goals and can lead to poor robustness and suboptimal performance. Tools that facilitate the analysis and characterization of the effects of uncertainties enable designers to develop more robustly performing systems. The need to analyze the effects of uncertainty is particularly acute when designing dynamical systems. Ultimately, if a robust system design is to be achieved, uncertainties must be accounted for up-front during the design process.

This section presents the novel parametric robust design optimization (RDO) framework that treats uncertain dynamical systems described by linear or nonlinear ordinary differential equations (ODEs). System uncertainties, such as parameters, initial conditions, sensor/actuator noise, or forcing functions, are modeled using Generalized Polynomial Chaos (gPC) and are solved quantitatively using a least-square collocation method (LSCM). The computational efficiencies gained by gPC and LSCM enable the inclusion of uncertainty statistics in the optimization process.

It was found that the benefits of treating uncertainty during the parametric design optimization process are most evident when active constraints are present; therefore, particular attention is given to its use in a constraint-based formulation of *multi-objective optimization* (cMOO). These benefits are illustrated in an optimal vehicle suspension design case-study where the opposing performance criteria related to passenger ride comfort, suspension operating displacement, and road holding are simultaneously accounted for.

6.2 Parametric Optimal Design of Deterministic ODE Systems

Given the Euler-Lagrange EOMs defined in Section 2.8*,* the NLP-based formulation of the deterministic optimal design problem can be described as,

$$
\begin{aligned}
\min_{\boldsymbol{x}} \quad & \mathrm{J} = \mathrm{J_{obj}} \\
\text{s.t.} \quad & \textbf{Dynamics} \\
& \boldsymbol{\mathcal{F}}(\boldsymbol{q}, \boldsymbol{v}, \dot{\boldsymbol{v}}, \boldsymbol{\theta}) = \boldsymbol{\tau} \\
& \textbf{Kinematics} \\
& \dot{\boldsymbol{q}} = \boldsymbol{H}(\boldsymbol{q}, \boldsymbol{\theta})\boldsymbol{v} \\
& \textbf{Outputs} \\
& \boldsymbol{y} = \boldsymbol{\mathcal{O}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\theta}) \\
& \textbf{Constraints} \\
& \boldsymbol{\mathcal{C}}(\boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{\tau}) \leq \boldsymbol{0} \\
& \textbf{Hard ICs \& TCs Conditions} \\
& \boldsymbol{q}(0) = \boldsymbol{q}_0, \boldsymbol{q}(t_f) = \boldsymbol{q}_{t_f} \\
& \dot{\boldsymbol{q}}(0) = \dot{\boldsymbol{q}}_0, \dot{\boldsymbol{q}}(t_f) = \dot{\boldsymbol{q}}_{t_f}
\end{aligned} \tag{93}
$$

where $\mathrm{J} = \mathrm{J_{obj}}$ represents the design objective function; $\boldsymbol{\mathcal{C}}$ is the list of design constraints; $\boldsymbol{x}$ is the list of optimization variables; and $\{\boldsymbol{q}(0), \dot{\boldsymbol{q}}(0), \boldsymbol{q}(t_f), \dot{\boldsymbol{q}}(t_f)\}$ are the system's initial conditions (ICs) and optional terminal conditions (TCs).

As discussed in Section 3.3, (93) may be approached from either a SeqNLP or SimNLP perspective. Also, the DS class of optimization solvers—techniques such as Genetic Algorithms, Differential Evolution, and Particle Swarm—typically only treat unconstrained optimization problems. Therefore, all the inequality design constraints in (93) need to be converted from *hard* constraints to *soft* constraints using the techniques presented in Section 3.4. Once the inequality constraints have been converted to penalty terms, equation (93) can be reformulated as,

$$
\begin{aligned}
\min_{\boldsymbol{x}} \quad & \mathrm{J} = \mathrm{J_{obj}} + \mathrm{J_{constr}} \\
\text{s.t.} \quad & \textbf{Dynamics} \\
& \boldsymbol{\mathcal{F}}\big(\boldsymbol{q}(t), \boldsymbol{v}(t), \dot{\boldsymbol{v}}(t), \boldsymbol{\theta}(t)\big) = \boldsymbol{\tau}(t) \\
& \textbf{Kinematics} \\
& \dot{\boldsymbol{q}}(t) = \boldsymbol{H}\big(\boldsymbol{q}(t), \boldsymbol{\theta}(t)\big)\boldsymbol{v}(t) \\
& \textbf{Outputs} \\
& \boldsymbol{y}(t) = \boldsymbol{\mathcal{O}}\big(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t), \boldsymbol{\theta}(t)\big) \\
& \textbf{Initial Conditions} \\
& \boldsymbol{q}(0) = \boldsymbol{q}_0, \dot{\boldsymbol{q}}(0) = \dot{\boldsymbol{q}}_0
\end{aligned} \tag{94}
$$

where the equality constraints from the continuous dynamics are implicit in the calculation of the objective function. Approaching (94) from the SeqNLP perspective allows the DS class of unconstrained solvers to be used.

Ultimately, the design task encoded in (93)–(94) is to determine what values of the manipulated variables $\boldsymbol{x}$ minimize J.

6.3 Parametric Optimal Design of Uncertain ODE Systems

The new framework for RDO of uncertain dynamical systems is now presented; this is a reformulation of (93) where IC, sensor, actuator, and parameter uncertainties are treated in a unified manner through the gPC techniques described in Section 4.2. The reformulation is,

$$\begin{array}{ll} \min\limits_{\boldsymbol{x}} & \mathrm{J}_{\mathrm{obj}}(t;\boldsymbol{\xi}) \\ \mathrm{s.t.} & \textbf{Dynamics} \\ & \boldsymbol{\mathcal{F}}\big(\boldsymbol{q}(t;\boldsymbol{\xi}),\boldsymbol{v}(t;\boldsymbol{\xi}),\dot{\boldsymbol{v}}(t;\boldsymbol{\xi}),\boldsymbol{\theta}(t;\boldsymbol{\xi})\big)=\boldsymbol{\tau}(t;\boldsymbol{\xi}) \\ & \textbf{Kinematics} \\ & \dot{\boldsymbol{q}}(t;\boldsymbol{\xi})=\boldsymbol{H}\big(\boldsymbol{q}(t;\boldsymbol{\xi}),\boldsymbol{\theta}(t;\boldsymbol{\xi})\big)\boldsymbol{v}(t;\boldsymbol{\xi}) \\ & \textbf{Outputs} \\ & \boldsymbol{y}(t;\boldsymbol{\xi})=\boldsymbol{\mathcal{O}}\big(\boldsymbol{q}(t;\boldsymbol{\xi}),\dot{\boldsymbol{q}}(t;\boldsymbol{\xi}),\boldsymbol{\theta}(t;\boldsymbol{\xi})\big) \\ & \textbf{Constraints} \\ & \boldsymbol{\mathcal{C}}(\boldsymbol{y}(t;\boldsymbol{\xi}),\boldsymbol{\theta}(t;\boldsymbol{\xi}),\boldsymbol{\tau}(t;\boldsymbol{\xi}),t)\leq\boldsymbol{0} \\ & \textbf{Hard ICs \& TCs Conditions} \\ & \boldsymbol{q}(0;\boldsymbol{\xi})=\boldsymbol{q}_0,\boldsymbol{q}\big(t_f;\boldsymbol{\xi}\big)=\boldsymbol{q}_{t_f} \\ & \dot{\boldsymbol{q}}(0;\boldsymbol{\xi})=\dot{\boldsymbol{q}}_0,\dot{\boldsymbol{q}}\big(t_f;\boldsymbol{\xi}\big)=\dot{\boldsymbol{q}}_{t_f} \end{array} \tag{95}$$

The most interesting part of the new design framework comes in the ability to approach the design accounting for uncertainties by way of statistical moments of $\boldsymbol{\xi}$**,** such as expected values, variances, or standard deviations. These statistical moments may now be included in the definitions of the objective function, $\mathrm{J}(t;\boldsymbol{\xi})$, and constraint equations, $\boldsymbol{\mathcal{C}}(t;\boldsymbol{\xi})$. From [1], the statistical expected value is defined as,

$$\mu_x = E[x(\xi)] = \int_{\boldsymbol{\Omega}} x(\xi)\, f(\xi) d\xi \tag{96}$$

and the variance,

$$\sigma_x^2 = Var[x(\xi)] = \int_{\mathbf{\Omega}} (x(\xi) - \mu_x)^2 f(\xi) d\xi = E[(x(\xi) - \mu_x)^2] \tag{97}$$

with the standard deviation, $\sigma_x = \sqrt{Var[x(\xi)]}$. With these definitions new objective function terms may be defined. For example, the mean and standard deviation of an output may be efficiently computed by,

$$\mu_{\boldsymbol{y}} = E[\boldsymbol{y}(\mathrm{t}; \boldsymbol{\xi})] = \boldsymbol{y}^0(\mathrm{t})\langle \Psi^0, \Psi^0 \rangle \tag{98}$$

$$\sigma_{\boldsymbol{y}} = \sqrt{E\left[\left(\boldsymbol{y}(\mathrm{t}; \boldsymbol{\xi}) - \mu_{\boldsymbol{y}}\right)^2\right]} = \sqrt{\sum_{j=1}^{n_b} (\boldsymbol{y}^j(\mathrm{t}))^2 \langle \Psi^j, \Psi^j \rangle} \tag{99}$$

Notice that due to the orthogonality of the polynomial basis these computations result in a reduced set of arithmetic operations on the respective expansion coefficients. Also, recall that $\langle \Psi^j, \Psi^j \rangle = 1, \forall j$ when using *normalized basis*; *standardized basis* are constant and may be computed off-line for efficiency using (12). A number of efficient statistical quantities may be determined from the expansion coefficients. Examples of these statistical terms were presented in Section 5.4 for the motion planning problem and additional terms are defined within the context of the case-study detailed in Section 7.4.

Equation (95) is the NLP formulation of the new framework for the optimal design of uncertain dynamical systems; it may also be solved through a SeqNLP or SimNLP approach as described in Section 6.2. The SeqNLP approach directly leverages the LSCM-based gPC solver, however, the SimNLP approach requires slight modification in the formulation to account for the full discretization of (3)–(5) in light of the LSCM technique.

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \mathrm{J_{obj}}(t; \boldsymbol{\xi}) \\ \mathrm{s.t.} \quad & \textbf{Dynamics } ({}_1\boldsymbol{\mu}) \\ & \boldsymbol{\mathcal{F}}\left(\boldsymbol{q}({}_1\boldsymbol{\mu}), \boldsymbol{v}({}_1\boldsymbol{\mu}), \dot{\boldsymbol{v}}({}_1\boldsymbol{\mu}), \boldsymbol{\theta}({}_1\boldsymbol{\mu})\right) = \boldsymbol{\tau}({}_1\boldsymbol{\mu}) \\ & \textbf{Kinematics } ({}_1\boldsymbol{\mu}) \\ & \dot{\boldsymbol{q}}({}_1\boldsymbol{\mu}) = \boldsymbol{H}\left(\boldsymbol{q}({}_1\boldsymbol{\mu}), \boldsymbol{\theta}({}_1\boldsymbol{\mu})\right) \boldsymbol{v}({}_1\boldsymbol{\mu}) \\ & \textbf{Outputs } ({}_1\boldsymbol{\mu}) \\ & \boldsymbol{y}({}_1\boldsymbol{\mu}) = \boldsymbol{\mathcal{O}}\left(\boldsymbol{q}({}_1\boldsymbol{\mu}), \dot{\boldsymbol{q}}({}_1\boldsymbol{\mu}), \boldsymbol{\theta}({}_1\boldsymbol{\mu})\right) \\ & \textbf{Initial Conditions } ({}_1\boldsymbol{\mu}) \end{aligned} \tag{100}$$

$$\boldsymbol{q}(0;\ {}_{1}\boldsymbol{\mu}) = \boldsymbol{q}_{0,1}, \dot{\boldsymbol{q}}(0;\ {}_{1}\boldsymbol{\mu}) = \dot{\boldsymbol{q}}_{0,1}$$

$$\vdots$$

Dynamics (${}_{n_{cp}}\boldsymbol{\mu}$)

$$\boldsymbol{\mathcal{F}}\left(\boldsymbol{q}\left({}_{n_{cp}}\boldsymbol{\mu}\right), \boldsymbol{v}\left({}_{n_{cp}}\boldsymbol{\mu}\right), \dot{\boldsymbol{v}}\left({}_{n_{cp}}\boldsymbol{\mu}\right), \boldsymbol{\theta}\left({}_{n_{cp}}\boldsymbol{\mu}\right)\right) = \boldsymbol{\tau}\left({}_{n_{cp}}\boldsymbol{\mu}\right)$$

Kinematics (${}_{n_{cp}}\boldsymbol{\mu}$)

$$\dot{\boldsymbol{q}}\left({}_{n_{cp}}\boldsymbol{\mu}\right) = \boldsymbol{H}\left(\boldsymbol{q}\left({}_{n_{cp}}\boldsymbol{\mu}\right), \boldsymbol{\theta}\left({}_{n_{cp}}\boldsymbol{\mu}\right)\right)\boldsymbol{v}\left({}_{n_{cp}}\boldsymbol{\mu}\right)$$

Outputs (${}_{n_{cp}}\boldsymbol{\mu}$)

$$\boldsymbol{y}\left({}_{n_{cp}}\boldsymbol{\mu}\right) = \boldsymbol{\mathcal{O}}\left(\boldsymbol{q}\left({}_{n_{cp}}\boldsymbol{\mu}\right), \dot{\boldsymbol{q}}\left({}_{n_{cp}}\boldsymbol{\mu}\right), \boldsymbol{\theta}\left({}_{n_{cp}}\boldsymbol{\mu}\right)\right)$$

Initial Conditions (${}_{n_{cp}}\boldsymbol{\mu}$)

$$\boldsymbol{q}\left(0; {}_{n_{cp}}\boldsymbol{\mu}\right) = \boldsymbol{q}_{0,n_{cp}}, \dot{\boldsymbol{q}}\left(0; {}_{n_{cp}}\boldsymbol{\mu}\right) = \dot{\boldsymbol{q}}_{0,n_{cp}}$$

Constraints

$$\boldsymbol{\mathcal{C}}(\boldsymbol{y}(\boldsymbol{\xi}), \boldsymbol{\theta}(\boldsymbol{\xi}), \boldsymbol{\tau}(\boldsymbol{\xi}), t) \leq \mathbf{0}$$

Equation (100) duplicates the deterministic dynamical equations (3)–(5) n_{cp} times where each set has a unique collocation point, ${}_{k}\boldsymbol{\mu}$. Each unique set of dynamical equations is then fully discretized and $\boldsymbol{x}$ is updated appropriately as described in Section 6.2. However, the system constraints, $\boldsymbol{\mathcal{C}}(\boldsymbol{y}(\boldsymbol{\xi}), \boldsymbol{\theta}(\boldsymbol{\xi}), \boldsymbol{\tau}(\boldsymbol{\xi}), t) \leq \mathbf{0}$, are calculated using the statistical properties determined by the LSCM and the n_{cp} sets of dynamical equations.

If a DS solver is to be used then (95) must be reformulated as an unconstrained problem. This requires all the *hard* design constraints in (95) to be converted to *soft* constraints by including them in the objective function. This may be accomplished using the techniques presented in Section 3.4. Once the constraints have been converted to penalty terms, equation (95) can be reformulated as an unconstrained optimization problem by,

$$\min_{\boldsymbol{x}} \quad \mathrm{J} = \mathrm{J}_{\mathrm{obj}}(t; \boldsymbol{\xi}) + \mathrm{J}_{\mathrm{const}}(\mathrm{t}; \boldsymbol{\xi}) \qquad (101)$$

$$\text{s.t.}\quad \begin{array}{l} \textbf{Dynamics} \\ \boldsymbol{\mathcal{F}}\big(\boldsymbol{q}(t;\boldsymbol{\xi}),\boldsymbol{v}(t;\boldsymbol{\xi}),\dot{\boldsymbol{v}}(t;\boldsymbol{\xi}),\boldsymbol{\theta}(t;\boldsymbol{\xi})\big)=\boldsymbol{\tau}(t;\boldsymbol{\xi}) \\ \textbf{Kinematics} \\ \dot{\boldsymbol{q}}(t;\boldsymbol{\xi})=\boldsymbol{H}\big(\boldsymbol{q}(t;\boldsymbol{\xi}),\boldsymbol{\theta}(t;\boldsymbol{\xi})\big)\boldsymbol{v}(t;\boldsymbol{\xi}) \\ \textbf{Outputs} \\ \boldsymbol{y}(t;\boldsymbol{\xi})=\boldsymbol{\mathcal{O}}\big(\boldsymbol{q}(t;\boldsymbol{\xi}),\dot{\boldsymbol{q}}(t;\boldsymbol{\xi}),\boldsymbol{\theta}(t;\boldsymbol{\xi})\big) \\ \textbf{Initial Conditions} \\ \boldsymbol{q}(0;\boldsymbol{\xi})=\boldsymbol{q}_0,\dot{\boldsymbol{q}}(0;\boldsymbol{\xi})=\dot{\boldsymbol{q}}_0 \end{array}$$

Equation (101) is analogous to the SeqNLP version defined in (95); it is dependent on the implicit ODE integration of the dynamics found in (3)–(5) and their associated ICs.

The new framework presented in (95), (100)–(101) allows designers to directly treat the effects of modeled uncertainties during the optimal design process. The computational efficiencies of gPC enable the inclusion of statistical measures in objective function and constraint equations at a reduced computational cost as compared to contemporary techniques. However, the framework does introduce an additional layer of modeling and computation [79]. Therefore, it is of value to ask when the application of the new framework (95), (100)–(101) will yield a more robust design over the traditional deterministic optimal design approach presented in (93)–(94). Based on the authors' experience, the following general guidelines can help determine if a given design will benefit from the new framework:

1. **System Nonlinearities:** The probability density functions (PDFs) of uncertainties are likely to become skewed when propagated through a nonlinear system; the new framework will capture this information in the expected value measures.
2. **Active Constraints:** Any design that has active constraints (i.e., $\mathcal{C}_i(t;\boldsymbol{\xi})=0$ for at least one *i*) will benefit from the inclusion of standard deviation information in the constraint definitions; for example, $\mathcal{C}_i(\mathrm{t};\boldsymbol{\xi})=\big(\mu_{y_i}\pm\sigma_{y_i}\big)-\overline{y}_i=0$ will off-set the optimal design in a standard deviation sense to account for the entire family of realizable systems.
3. **Multi-objective Problems:** Any multi-objective design that is reformulated as a cMOO will benefit from the added statistical information of the new framework. This results in a shifted, or offset, Pareto optimal set (as will be illustrated in the case-study presented in Section 6.4). Application of the new framework directly to a penalty-based MOO formulated problem will likely not capture the Pareto offset determined by the cMOO formulation. This point is simply a restating of item #2 within the MOO context.

The new framework is a general formulation for the optimal design of dynamical systems described by ODEs. In an effort to show-case the benefits of the new framework a vehicle suspension optimal design problem is presented next.

6.4 An Illustrative Case-Study

This section presents an optimal design case-study of a passive nonlinear vehicle suspension as an illustration of the benefits that the general framework presented in Section 6.3 can provide. Many studies related to the optimal design of vehicle suspension parameters are found in the literature [90, 92, 93, 95-97]. Studies include linear and nonlinear vehicle models; MOO design; passive, semi-active, and active suspensions; and uncertain road inputs. This case-study is not comprehensive, but aims to illustrate the benefits of the new framework. As such, a nonlinear quarter-car suspension model was selected that is subject to parameter uncertainties. The literature frequently accounts for three conflicting objectives in a MOO design setting: the passenger comfort (*ride*); suspension displacement (*rattle*); and tire road holding forces (*holding*). These opposing objective terms yield the expected Pareto optimal set for a given parameter set; this case-study will address the optimal design through a cMOO.

6.4.1 Vehicle suspension model

An idealized two degree-of-freedom (DOF) nonlinear quarter-car suspension model shown in Figure 6.1—An idealized 2-DOF deterministic quarter-car suspension model with a nonlinear asymmetric damper was used.

This system results in the following deterministic nonlinear dynamical equations-of-motion (EOMs),

$$\begin{aligned} \ddot{z}_s &= -\frac{k_s}{m_s}(z_s - z_u) - \frac{1}{m_s}\mathcal{D}(\dot{z}_s, \dot{z}_u) \\ \ddot{z}_u &= \frac{k_s}{m_u}(z_s - z_u) + \frac{1}{m_u}\mathcal{D}(\dot{z}_s, \dot{z}_u) - \frac{k_u}{m_u}(z_u - z_g) \end{aligned} \tag{102}$$

The model has sprung and unsprung masses, m_s and m_u; vertical mass positions about the equilibrium, $\{z_s, z_u\}$, and velocities $\{\dot{z}_s, \dot{z}_u\}$; suspension spring and damping coefficients, k_s and

b_s; tire spring coefficient, k_u; and ground input position, z_g. The system is nonlinear due to the asymmetric damping force that is dependent on the velocity direction.

$$\mathcal{D}(\dot{z}_s, \dot{z}_u) = \begin{cases} b_s(\dot{z}_s - \dot{z}_u), & (\dot{z}_s - \dot{z}_u) \geq 0 \\ \eta\, b_s(\dot{z}_s - \dot{z}_u), & (\dot{z}_s - \dot{z}_u) \geq 0 \end{cases} \tag{103}$$

The ratio of damping forces is determined by the scalar η.

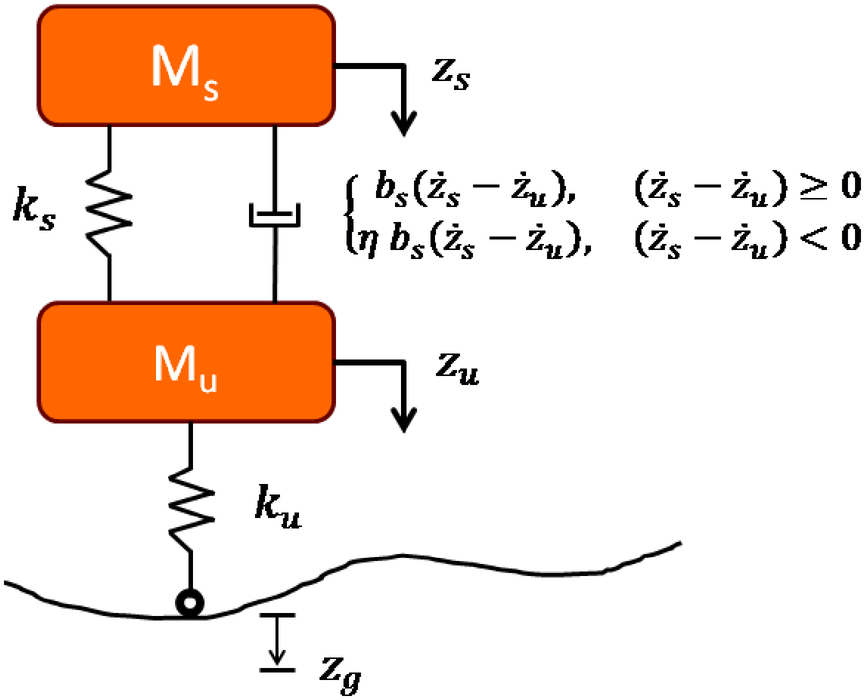
M_s
z_s
k_s
$\begin{cases} b_s(\dot{z}_s - \dot{z}_u), & (\dot{z}_s - \dot{z}_u) \geq 0 \\ \eta\, b_s(\dot{z}_s - \dot{z}_u), & (\dot{z}_s - \dot{z}_u) < 0 \end{cases}$
M_u
z_u
k_u
z_g

Figure 6.1—An idealized 2-DOF deterministic quarter-car suspension model with a nonlinear asymmetric damper

The literature contains various methods for modeling the road input, z_g. A number of authors used stationary ergodic Gaussian inputs for linear quarter-car models through a power-spectral density (PSD) transformation of the system's linear frequency response [92, 95]. Additional attention was given to frequency weighted power-spectrum inputs based on standards such as ISO 2631 [90, 92, 95, 96]. This approach directly accounts for uncertainty in the road input of a linear system. Verros used the same Gaussian uncertain inputs for nonlinear quarter-car models through application of a Monte Carlo sampling technique [90]. These are examples of continuous road irregularity inputs. Additional authors treated isolated road irregularities such as speed bumps and potholes [96, 97]. These inputs were modeled by,

$$z_g = A\, sin(\omega\, t) \tag{104}$$

where A represents the amplitude of the bump or pothole; $\omega = \pi\, v/l$ is the frequency of the irregularity determined by the vehicle velocity v and base length of the irregularity l; and t represents time over a finite time interval.

This work chose to use a series of isolated road bumps defined by (104). Each bump is uniquely spaced with no overlap with one another and their amplitude was $A = 0.15$ meters. The frequencies of the speed bumps were selected to be $\omega = [1,\ 5, 10, 15]$ Hertz. Filtered Gaussian noise with a maximum amplitude of $A = 0.03$ meters was super-imposed over the series of speed bumps. The cut-off frequency of the filtered Gaussian noise was 35 Hertz. A representative road input signal is shown in Figure 6.2.

6.4.2 Optimal Design of Deterministic System

Contemporary optimal designs of a vehicle suspension commonly account for three opposing performance indexes: *ride*, *rattle*, and road *holding*. These performance indexes may be defined as,

$$J_{ride} = \int_0^{\mathrm{t_f}} \ddot{z}_s^2\, dt \tag{105}$$

$$J_{rattle} = \int_0^{\mathrm{t_f}} (z_s - z_u)^2\, dt \tag{106}$$

$$J_{holding} = \int_0^{\mathrm{t_f}} \left(z_g - z_u\right)^2 dt \tag{107}$$

The *ride* index aims to minimize the vertical accelerations experienced by a passenger. The *rattle* index aims to avoid the suspension displacement reaching its physical limits. The *holding* aims to minimize the variation of the dynamic force between the tire and the road [95]. Equation (107) is defined by the tire deflection. Given the linear relationship between the tire deflection and the tire/road force the same minimized variation is accomplished.

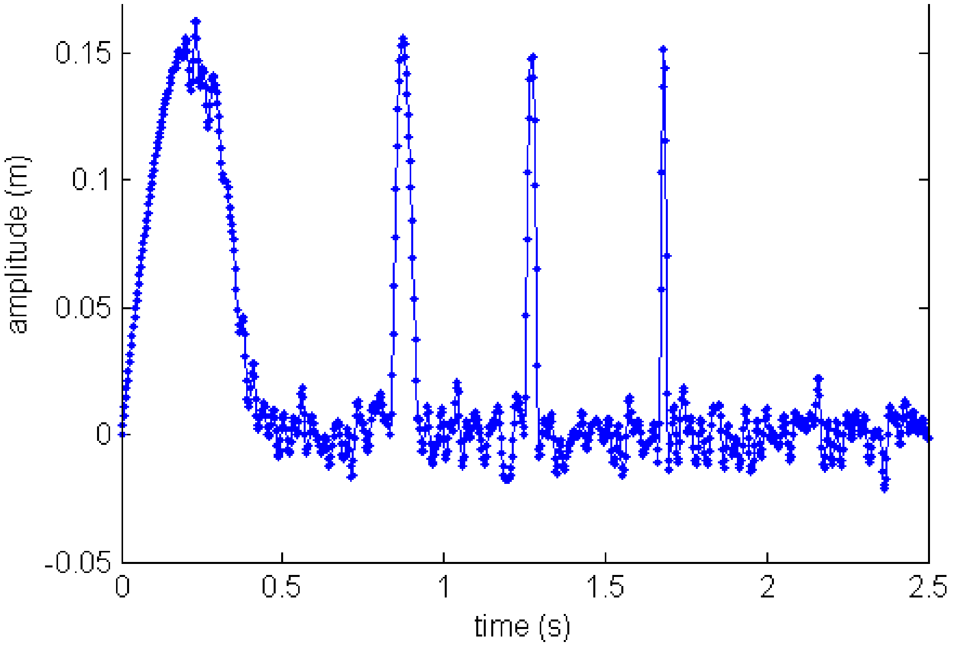

Figure 6.2—A representative road input signal created with a series of isolated speed bumps with filtered noise superimposed.

A MOO approach to treating these opposing objectives is to define a scalarized cost function as,

$$J = w_1 J_{ride} + w_2 J_{rattle} + w_3 J_{holding} \tag{108}$$

Such a definition results in the Pareto optimal set which depends on the weights $\{w_1, w_2, w_3\}$. As discussed in Section 6.2, the MOO can be reformulated in a constraint formulation yielding a similar Pareto optimal set. For example, J_{rattle} and $J_{holding}$ can be converted to *hard* constraints of the dynamic optimization problem. However, to reflect a more physical meaning of these qualities, (105)–(106) will be slightly redefined to root-mean-square (*rms)* values determined over the trajectory of the system.

$$J_{rattle} = rms(z_s - z_u) = \sqrt{\frac{1}{\mathrm{t_f}} \int_0^{\mathrm{t_f}} (z_s - z_u)^2 \, dt} \tag{109}$$

$$J_{holding} = rms(z_g - z_u) = \sqrt{\frac{1}{\mathrm{t_f}} \int_0^{\mathrm{t_f}} (z_g - z_u)^2 \, dt} \tag{110}$$

Hard constraints of the form presented in (10) and based on(109)–(110) can be defined as,

$$\underline{J}_{rattle} \leq J_{rattle} \leq \bar{J}_{rattle} \tag{111}$$

$$J_{holding} - \bar{J}_{holding} \le 0 \tag{112}$$

where $\left\{\underline{J}_{rattle}, \bar{J}_{rattle}, \bar{J}_{holding}\right\}$ represent lower/upper bounding constraints. By sweeping through reasonable ranges for $\left\{\underline{J}_{rattle}, \bar{J}_{rattle}, \bar{J}_{holding}\right\}$ a Pareto optimal set may be found.

Therefore, the NLP-based cMOO formulation for the deterministic vehicle suspension design problem is,

$$
\begin{aligned}
\min_{\boldsymbol{x}=\{k_s,b_s\}} \quad & \mathrm{J} = \int_0^{\mathrm{t_f}} \ddot{z}_s^2\,dt = y_5 \\
\text{s.t.} \quad & \textbf{Dynamics} \\
& \ddot{z}_s = -\frac{k_s}{m_s}(z_s - z_u) - \frac{1}{m_s}\mathcal{D}(\dot{z}_s,\dot{z}_u) \\
& \ddot{z}_s = \frac{k_s}{m_u}(z_s - z_u) + \frac{1}{m_u}\mathcal{D}(\dot{z}_s,\dot{z}_u) - \frac{k_u}{m_u}(z_u - z_g) \\
& \textbf{Outputs} \\
& \boldsymbol{y} = [z_s, \dot{z}_s, z_u, \dot{z}_u, \int_0^{\mathrm{t_f}} \ddot{z}_s{}^2\,dt] \\
& \textbf{Constraints} \\
& J_{rattle} - \bar{J}_{rattle} \le 0,\ \ (extension) \\
& \underline{J}_{rattle} - J_{rattle} \le 0,\ \ (compression) \\
& J_{holding} - \bar{J}_{holding} \le 0 \\
& \underline{k}_s \le k_s \le \bar{k}_s \\
& \underline{b}_s \le b_s \le \bar{b}_s \\
& \textbf{Initial Conditions} \\
& \boldsymbol{z}(0) = \boldsymbol{z}_0 \\
& \dot{\boldsymbol{z}}(0) = \dot{\boldsymbol{z}}_0
\end{aligned} \tag{113}
$$

where $\{\underline{k}_s, \bar{k}_s\}$ and $\{\underline{b}_s, \bar{b}_s\}$ are reasonable physical bounds on the spring and damping coefficients, respectively; $\{\boldsymbol{z}_0, \dot{\boldsymbol{z}}_0\}$ are initial conditions for the vector of state variables, $\boldsymbol{z} = [z_s, \dot{z}_s, z_u, \dot{z}_u]^T$; and the list of solver manipulated variables is $\boldsymbol{x} = \{k_s, b_s\}$. Table 6.1 lists all nominal values for the system parameters and bounds for (113). Notice that the fifth output, $y_5 = \int_0^{\mathrm{t_f}} \ddot{z}_s{}^2\,dt$, is equal to the defined measure for the *ride* quality to be minimized.

Equation (113) may also be reformulated for non-gradient-based DS solvers as was presented in (94).

Table 6.1—System parameters, bounds, and uncertainties

Parameter	Mean ($\boldsymbol{\mu}$)	Std ($\boldsymbol{\sigma}$)	Units (SI)
m_s	376	$m_s^0/8$	kg
m_u	$m_s/4$	-	kg
k_s	30,000	$k_s^0/8$	N/m
$\overline{k}_s$	300,000	-	N/m
$\underline{k}_s$	$\mathcal{F}(\overline{J}_{rattle}) = 23{,}339$	-	N/m
b_s	2,000	$b_s^0/8$	N-s/m
$\overline{b}_s$	50,000	-	N-s/m
$\underline{b}_s$	1+ σ_{b_s}	-	N-s/m
η	1.39	$\eta^0/8$	-
k_u	200,000	$k_u^0/8$	N/m
$\overline{J}_{holding}$	Figure 6.5	-	m
$\underline{J}_{rattle}, \overline{J}_{rattle}$	Figure 6.4	-	m
$\{\boldsymbol{z}_0, \dot{\boldsymbol{z}}_0\}$	zeros(n_p, 1)	-	-
v	11.18	-	m/s

6.4.3 Optimal Design of Uncertain Systems

None of the cited works for optimal vehicle suspension design treated uncertain system parameters; however, the new framework presented in Section 6.3 is capable of treating system uncertainties originating from sensor outputs, actuator inputs, as well as system parameters and initial conditions within the unified gPC methodology (as presented in Section 4.2). Given the prior emphasis on input uncertainty in the literature, this work focuses on illustrating treatment of parametric uncertainty applied to a nonlinear system model. Varying passenger and cargo loads, fatiguing/deteriorating suspension components, and variations in tire air pressure are all very practical sources of uncertainty in a vehicle. Therefore, five system parameters were selected for this study, $\boldsymbol{\theta}(\boldsymbol{\xi}) = \{m_s(\boldsymbol{\xi}), k_s(\boldsymbol{\xi}), b_s(\boldsymbol{\xi}), \eta(\boldsymbol{\xi}), k_u(\boldsymbol{\xi})\}$. Each uncertain parameter is assumed to have a uniform distribution and is therefore modeled with a Legendre polynomial expansion. This takes the form of,

$$\theta_r(\xi_r) = \theta_r^0 + \theta_r^1 \xi_r, \qquad r = 1 \ldots n_p \tag{114}$$

The corresponding parameter values are listed in Table 6.1. Figure 6.3 illustrates the uncertain nonlinear model.

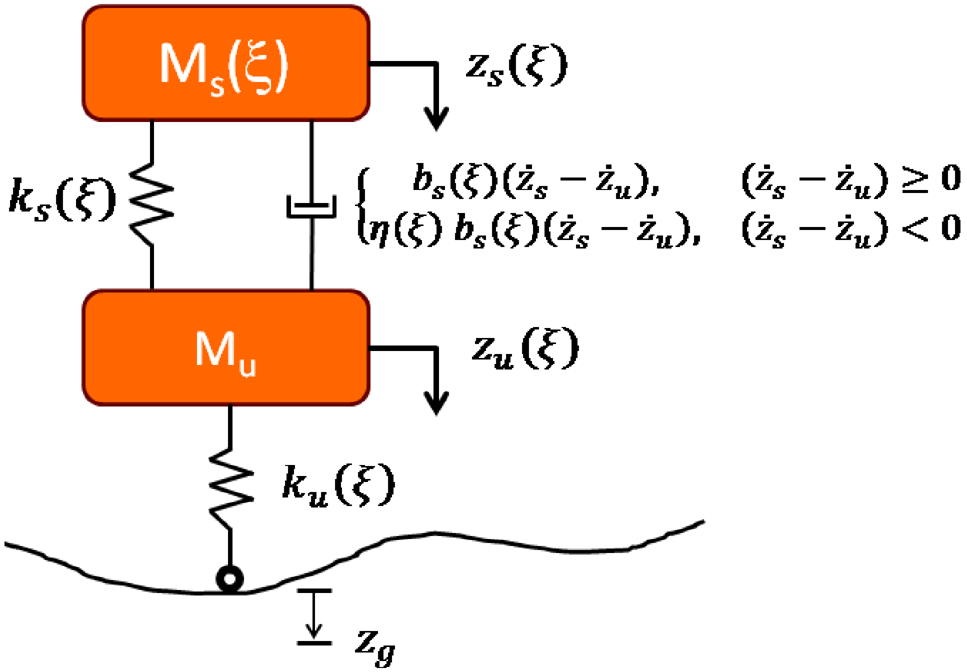
$M_s(\xi)$
$z_s(\xi)$
$k_s(\xi)$
$\begin{cases} b_s(\xi)(\dot{z}_s-\dot{z}_u), & (\dot{z}_s-\dot{z}_u)\ge 0 \\ \eta(\xi)\, b_s(\xi)(\dot{z}_s-\dot{z}_u), & (\dot{z}_s-\dot{z}_u)<0 \end{cases}$
M_u
$z_u(\xi)$
$k_u(\xi)$
z_g

Figure 6.3—An uncertain 2-DOF quarter-car suspension model with a nonlinear asymmetric damper. The five uncertain parameters are, $\boldsymbol{\theta}(\xi) = \{\boldsymbol{m_s}(\xi), \boldsymbol{k_s}(\xi), \boldsymbol{b_s}(\xi), \boldsymbol{\eta}(\xi), \boldsymbol{k_u}(\xi)\}$.

This system results in the following set of uncertain nonlinear EOMs,

$$\begin{aligned}\ddot{z}_s(\xi) &= -\frac{k_s(\xi)}{m_s(\xi)}\big(z_s(\xi)-z_u(\xi)\big)-\frac{1}{m_s(\xi)}\mathcal{D}\big(\dot{z}_s(\xi),\dot{z}_u(\xi)\big)\\ \ddot{z}_u(\xi) &= \frac{k_s(\xi)}{m_u}\big(z_s(\xi)-z_u(\xi)\big)+\frac{1}{m_u}\mathcal{D}\big(\dot{z}_s(\xi),\dot{z}_u(\xi)\big)-\frac{k_u(\xi)}{m_u}\big(z_u(\xi)-z_g(\xi)\big)\end{aligned} \tag{115}$$

and the corresponding NLP-based cMOO design problem is,

$$\begin{aligned} \min_{\boldsymbol{x}=\{k_s^0, b_s^0\}} \quad & \mathrm{J}(\boldsymbol{\xi}) \\ \text{s.t.} \quad & \textbf{Dynamics} \\ & \ddot{z}_s(\boldsymbol{\xi}) = -\frac{k_s(\boldsymbol{\xi})}{m_s(\boldsymbol{\xi})}\big(z_s(\boldsymbol{\xi})-z_u(\boldsymbol{\xi})\big)-\frac{1}{m_s(\boldsymbol{\xi})}\mathcal{D}\big(\dot{z}_s(\boldsymbol{\xi}),\dot{z}_u(\boldsymbol{\xi})\big)\\ & \ddot{z}_u(\boldsymbol{\xi}) = \frac{k_s(\boldsymbol{\xi})}{m_u}\big(z_s(\boldsymbol{\xi})-z_u(\boldsymbol{\xi})\big)+\frac{1}{m_u}\mathcal{D}\big(\dot{z}_s(\boldsymbol{\xi}),\dot{z}_u(\boldsymbol{\xi})\big)\\ & \qquad\qquad -\frac{k_u(\boldsymbol{\xi})}{m_u}\big(z_u(\boldsymbol{\xi})-z_g\big)\end{aligned} \tag{116}$$

Outputs

$$\boldsymbol{y} = \begin{bmatrix} z_s(\boldsymbol{\xi}) \\ \dot{z}_s(\boldsymbol{\xi}) \\ z_u(\boldsymbol{\xi}) \\ \dot{z}_u(\boldsymbol{\xi}) \\ \int_0^{\mathrm{t_f}} \left(\ddot{z}_s(\boldsymbol{\xi})\right)^2 dt \\ rms\left(z_u(\boldsymbol{\xi}) - z_g\right) \end{bmatrix}$$

Constraints

$$J_{rattle}(\boldsymbol{\xi}) - \bar{J}_{rattle} \leq 0, \;\; (extension)$$

$$\underline{J}_{rattle} - J_{rattle}(\boldsymbol{\xi}) \leq 0, \;\; (compression)$$

$$J_{holding}(\boldsymbol{\xi}) - \bar{J}_{holding} \leq 0$$

$$\underline{k}_s \leq k_s^0(\boldsymbol{\xi}) \leq \bar{k}_s$$

$$\underline{b}_s \leq b_s^0(\boldsymbol{\xi}) \leq \bar{b}_s$$

Initial Conditions

$$\boldsymbol{z}(0) = \boldsymbol{z}_0, \; \dot{\boldsymbol{z}}(0) = \dot{\boldsymbol{z}}_0$$

where the uncertain asymmetric damping force is,

$$\mathcal{D}\left(\dot{z}_s(\boldsymbol{\xi}), \dot{z}_u(\boldsymbol{\xi})\right) = \begin{cases} b_s(\boldsymbol{\xi})\left(\dot{z}_s(\boldsymbol{\xi}) - \dot{z}_u(\boldsymbol{\xi})\right), & \left(\dot{z}_s(\boldsymbol{\xi}) - \dot{z}_u(\boldsymbol{\xi})\right) \geq 0 \\ \eta(\boldsymbol{\xi})\, b_s(\boldsymbol{\xi})\left(\dot{z}_s(\boldsymbol{\xi}) - \dot{z}_u(\boldsymbol{\xi})\right), & \left(\dot{z}_s(\boldsymbol{\xi}) - \dot{z}_u(\boldsymbol{\xi})\right) \geq 0 \end{cases} \tag{117}$$

The objective function is now a function of the uncertain *ride* comfort, (which is the fifth system output, $y_5(\xi) = \int_0^{\mathrm{t_f}} \left(\ddot{z}_s(\xi)\right)^2 dt$).

$$\begin{aligned} \mathrm{J}(\xi) &= \mathrm{E}[y_5(\xi)] + \sqrt{Var[y_5(\xi)]} \\ &= \upmu_{y_5} + \sigma_{y_5} \\ &= y_5^0 \langle \Psi^0, \Psi^0 \rangle + \sqrt{\sum_{j=1}^{n_b} \left(y_5^j\right)^2 \langle \Psi^j, \Psi^j \rangle} \end{aligned} \tag{118}$$

The *rattle* and *holding* constraints are also functions of the uncertainties. The uncertain *rattle* constraints may be defined as,

$$J_{rattle}(\xi) = \begin{cases} \left(\left(\upmu_{z_s} + a_1 \sigma_{z_s}\right) - \left(\upmu_{z_u} - a_2 \sigma_{z_u}\right)\right) - \bar{J}_{rattle} \leq 0 \\ \underline{J}_{rattle} - \left(\left(\upmu_{z_s} - a_3 \sigma_{z_s}\right) - \left(\upmu_{z_u} + a_4 \sigma_{z_u}\right)\right) \leq 0 \end{cases} \tag{119}$$

where the various μ and σ computations take the form shown in (118) and defined in (98)–(99); the constants a_i represent scaling factors of the standard deviation. These constraints represent the extreme *rattle* conditions from a standard deviation perspective. The uncertain *holding* constraint is a function of the sixth system output, $y_6(\xi) = rms\big(z_u(\xi) - z_g\big)$ and may be defined as,

$$\begin{aligned} J_{holding}(\xi) &= \left(\mathrm{E}[y_6(\xi)] + a_5\sqrt{Var[y_6(\xi)]}\right) - \bar{J}_{holding} \leq 0 \\ &= \mu_{y_6} + a_5\sigma_{y_6} - \bar{J}_{holding} \leq 0 \\ &= \left(y_6^0 \langle \Psi^0, \Psi^0 \rangle + a_5 \sqrt{\sum_{j=1}^{n_b} \left(y_6^j\right)^2 \langle \Psi^j, \Psi^j \rangle} \right) - \bar{J}_{holding} \leq 0 \end{aligned} \tag{120}$$

Notice that due to the orthogonality of the polynomial basis the computations in (118)–(120) result in a reduced set of efficient operations on the respective expansion coefficients. No integrals are required and the statistical computations are relatively efficient.

It is important to re-emphasize that equations (119)–(120) constrain the system in a standard deviation sense. This means a subset of the systems within the probability space can still not satisfy the constraints. In order to guarantee that all systems within the probability space will satisfy the constraints, equations (119)–(120) would need to be redefined such that the *supremum* and/or *infimum* statistics are used instead of the standard deviations; however, the *supremum* and *infimum* are very expensive to calculate. The scaling constants, a_i, may be used to 'tune' the design such that a desired percentage of the systems from the probability space will satisfy the constraints. This point is illustrated in greater detail in the following Results Section.

Finally, the design objective of (116) is to determine mean values for the suspension components, $\boldsymbol{x} = \{\, k_s^0, b_s^0\}$, that minimize the *ride* being subject to the *rattle* and *holding* constraints; where $\{\, k_s^0, b_s^0\}$ are bounded by $\{\underline{k}_s, \overline{k}_s\}$ and $\{\underline{b}_s, \overline{b}_s\}$, respectively. Table 6.1 details all the uncertain parameters and associated bounds used in the case-study.

6.5 Results

The results for this case study were generated by approximating the uniform distributions for the uncertain parameters and corresponding dynamic states and outputs with a third order, $p_o = 3$, Legendre polynomial series expansion. This corresponds to fifty-six basis terms, $n_b = 56$, where $3n_b$ collocation points were generated using a Hammersley low-discrepency sequence (LDS) before being solved by the LSCM.

Figure 6.4 and Figure 6.5 best illustrate the benefits of treating uncertainty during the optimal design process. These results clearly show that the presence of uncertainty in a system results in an off-set of the Pareto optimal design trade-off surface. Figure 6.4 is the 2D Pareto curve showing trade-off between the *ride* objective and the *rattle* constraint. Figure 6.5 shows a 2D Pareto curve perpendicular to that shown in Figure 6.4; this figure shows the trade-off between the *ride* objective and the *holding* constraint.

Both Figure 6.4 and Figure 6.5 show the Pareto curves flattening out at some point. This occurs when the system transitions from one active constraint to the other. Meaning, in Figure 6.4, the active constraint is the *rattle* constraint while the curve has a negative slope. However, when the slope flattens out the *holding* constraint is active. Since the *holding constraint* is constant in this plane the Pareto curve has a slope of zero. The same behavior is evident in Figure 6.5, however, the roles switch. First, the *holding* constraint is active and then the system transitions to the flat *rattle* constraint.

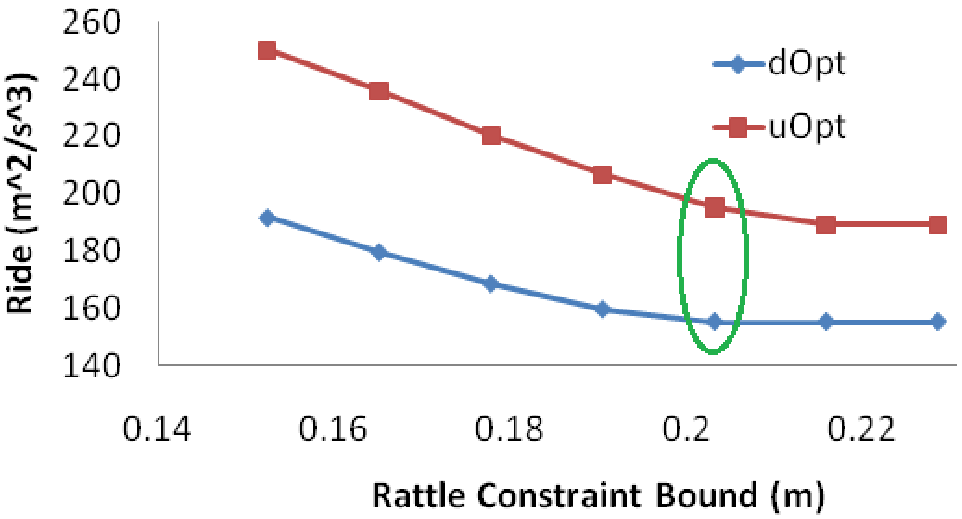

Figure 6.4—A single 2D plane from the 3D Pareto optimal set showing the trade-off between the objective *ride* and *rattle* constraint; the *holding* constraint is held constant, $\bar{J}_{holding} = 0.034\,[m]$. Both the deterministic (dOpt) and uncertain (uOpt) cases are shown. These results confirm that the presence of uncertainty requires an off-set of the Pareto optimal solution set to realize a more robust design. The set enclosed by the ellipse correspond to Figure 6.6.

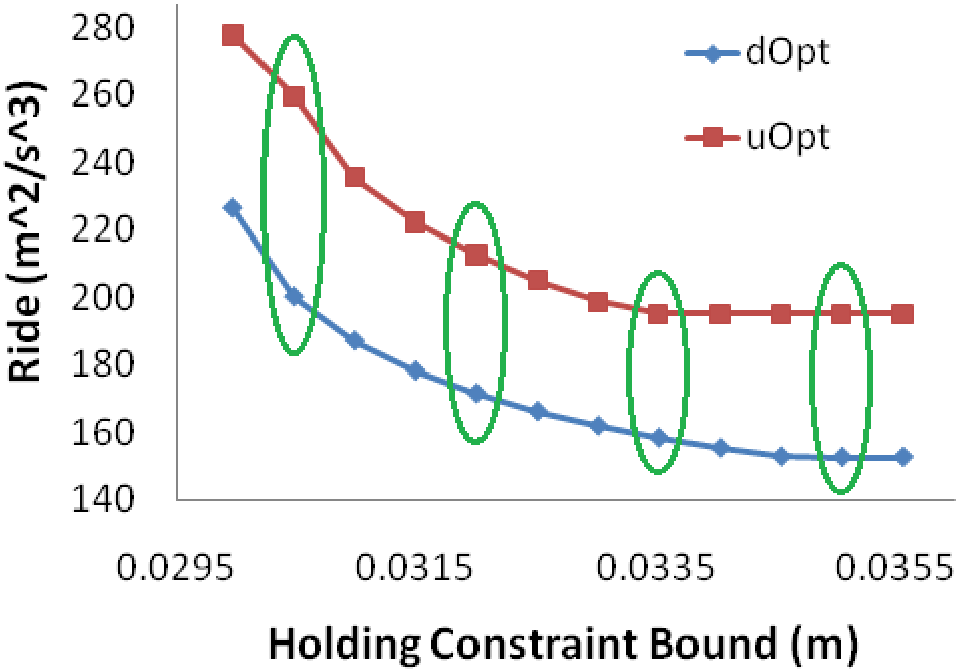

Figure 6.5—A single 2D plane from the 3D Pareto optimal set showing the trade-off between the objective *ride* and *holding* constraint; both the compression and extension *rattle* constraints are held constant, $J_{rattle} = \bar{J}_{rattle} = 0.203\,[m]$. Both the deterministic (dOpt) and uncertain (uOpt) cases are shown. These results confirm that the presence of uncertainty requires an off-set of the Pareto optimal solution set to realize a more robust design.

One distinct difference is apparent between Figure 6.4 and Figure 6.5; in Figure 6.4 the deterministic Pareto curve transitions its active constraints before the uncertain curve. However, Figure 6.5 has an opposite behavior; the uncertain curve transitions before the deterministic one. Figure 6.6 and Figure 6.7 help illustrate why this inconsistent behavior exists. Figure 6.6 presents a projection of the 3D optimal solution onto the three orthogonal 2D planes, $\{holding - rattle, holding - ride, rattle - ride\}$. When the bounding constraints are set to $\underline{J}_{rattle} = \bar{J}_{rattle} = 0.203\ [m]$ and $\bar{J}_{holding} = 0.034\ [m]$ the deterministic optimal design has an active *holding* constraint where the uncertain optimal design has an active *rattle* constraint. Since the uncertain σ^*_{rattle} is so large, the optimal design was pushed to a significantly lower $\mu^*_{holding}$ value when compared to the deterministic $x^*_{holding}$ value; thus, the deterministic Pareto transitions first. This same behavior is apparent in Figure 6.7 which shows the 2D projection only in $holding - rattle$ plane. Figure 6.5 presents the Pareto relating the *holding* constraint to the *ride* objective; so, as the *holding* bound increases Figure 6.7 shows how the uncertainty box enclosing the uncertain mean design—created by the standard deviation in the respective *rattle* and *holding* direction—encounters the *rattle* constraint before the corresponding deterministic solution does.

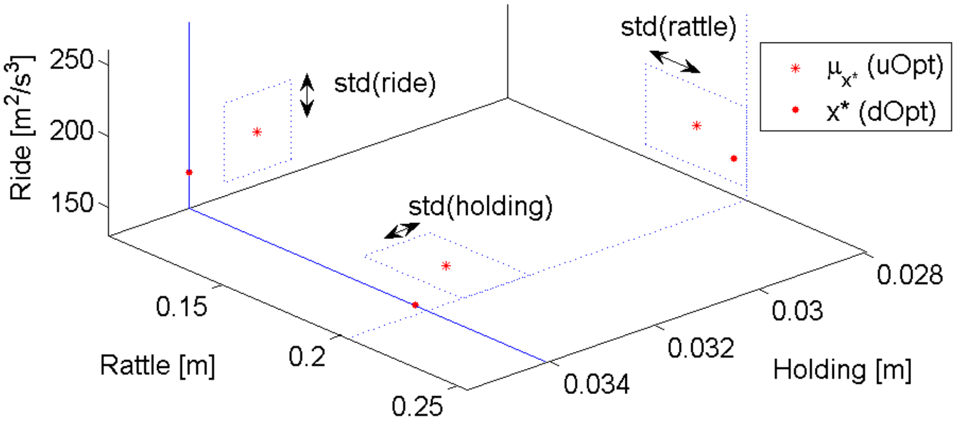

Figure 6.6—Projection of the 3D deterministic and uncertain solutions onto the three orthogonal 2D planes. This is an example of an optimal solution with an active *rattle* constraint. The constraint bounds are: $\underline{J}_{rattle} = \bar{J}_{rattle} = 0.203\ [m]$, and $\bar{J}_{holding} = 0.034\ [m]$.

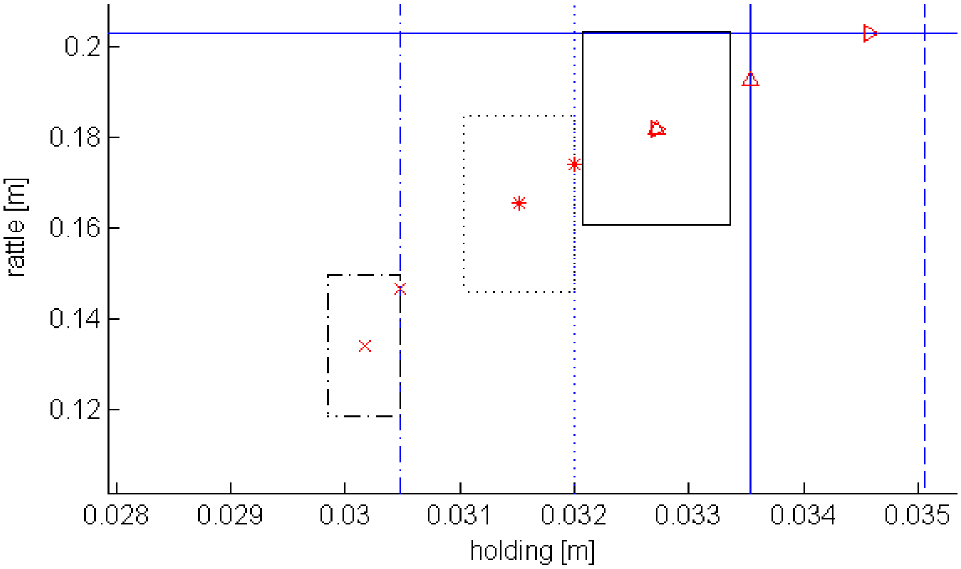

Figure 6.7—The 2D projection of Figure 6.5's 3D deterministic and uncertain solutions onto the *holding/rattle* plane. This shows the transition from an active *holding* constraint to an active *rattle* constraint as the *holding* bound is increased. The constraint bounds are: $\underline{J}_{rattle} = \bar{J}_{rattle} = 0.203\ [m]$, and $\bar{J}_{holding} = \{0.0305, 0.032, 0.0335, 0.035\}\ [m]$. The markers for the deterministic and uncertain mean designs correspond within a given set. Also, the line of the uncertainty box associated with a given set matches line of the *holding* bound for that set.

The corresponding 2D trade-off curve in the parameter space for the case corresponding to Figure 6.5 with $\underline{J}_{rattle} = \bar{J}_{rattle} = 0.203\ [m]$ is shown in Figure 6.8. This figure is instructive in that it shows the optimal mean spring constant, $(k_s^0)^*$, is found to be at the lower bound, $\underline{k}_s$, for a majority of the designs. Only when $\bar{J}_{holding}$ is equal to the two lowest values shown in Figure 6.5 does the optimal mean spring constant leave the lower bounding constraint. This behavior makes sense in that the *ride* objective is most influenced by the k_s versus b_s [89]; therefore, as long as the active constraint can be satisfied with $k_s^0 = \underline{k}_s$ then the Pareto curve is largely defined by the damping mean, b_s^0. Again, the design from the new framework shifts the optimal parameter set relative to that designed by a traditional deterministic optimal design.

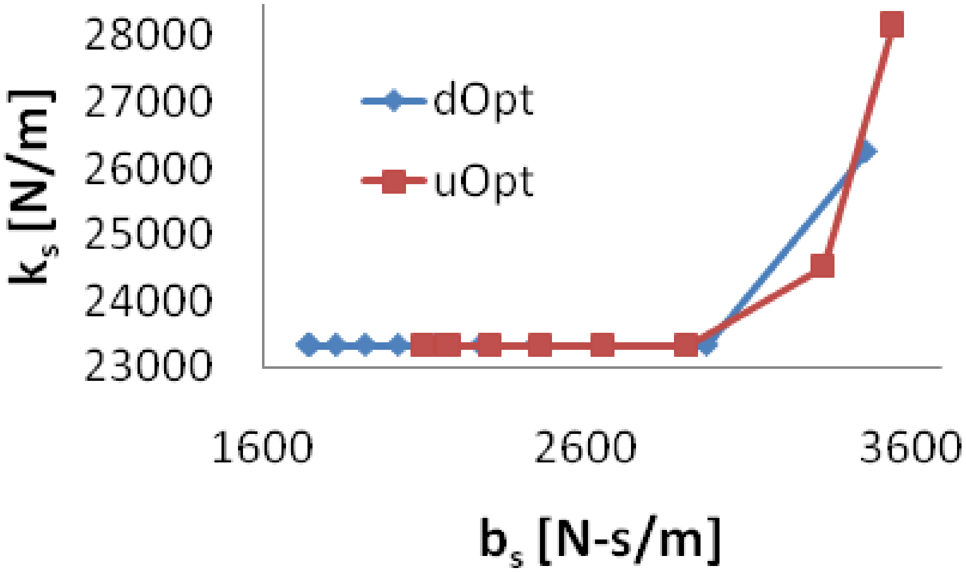

Figure 6.8—Parameter Pareto trade-off curve when $\underline{J}_{rattle} = \bar{J}_{rattle} = \mathbf{0.203}$ [$\boldsymbol{m}$].

The final point that illustrates the benefits of the new framework is shown in Figure 6.9. This figure shows the resulting *rattle* constraint values of the optimal deterministic design applied to an uncertain system. A 1000 sample Monte Carlo simulation shows that 59.6% of the systems violated the constraints. However, the design produced by the new framework defined in (116)–(120), with unity standard deviation scaling, $a = 1$, resulted in only 11.4% of the systems violating the constraints. Slightly increasing the standard deviation scaling to $a = 1.25$ resulting in only 3.5% of the systems failing to satisfy the constraints (see Figure 6.10). As described in Section 6.4.3, designers may choose to approach the problem with the constraints defined with *supremum* and/or *infimum* measures to guarantee the entire family of systems in the probability space will satisfy the constraints. However, this approach is relatively computationally costly. Therefore, a scaled standard deviation approach, as shown here, can yield an acceptably robust design. Similar design results are observed with regards to an active *holding* constraint as well, but are not shown for brevity.

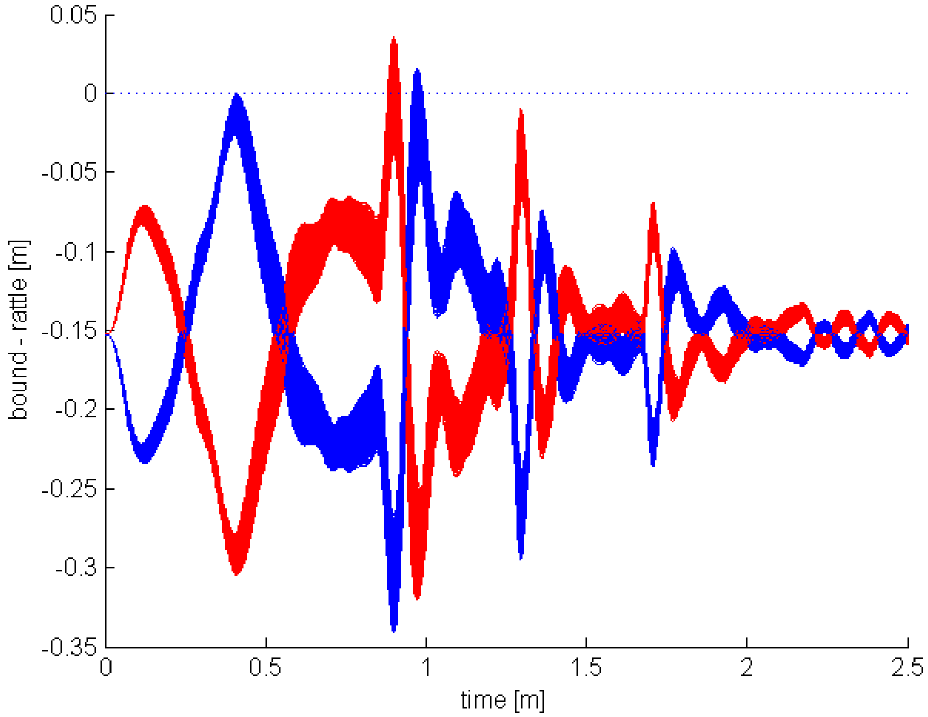

Figure 6.9—Monte Carlo results (1000 runs) showing 59.6% of the systems in the probability space violate the *rattle* constraints when the deterministic optimal design is applied to an uncertain system; where $\underline{J}_{rattle} = \bar{J}_{rattle} = 0.152\ [m]$ and $\bar{J}_{holding} = 0.034\ [m]$.

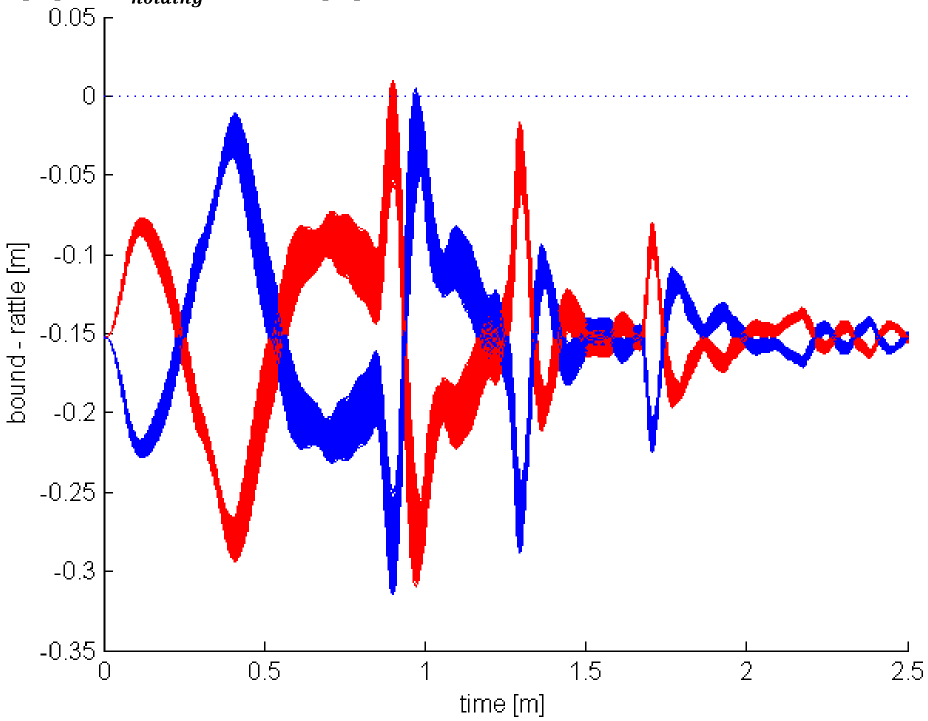

Figure 6.10—Constraint violations from designs produced with the new framework can be controlled, or tuned, with the proper selection of the standard deviation scaling. Slightly increasing the scaling from $a = 1$ to $a = 1.25$ reduces the number of systems violating the constraints from 11.4% to 3.5%; where 1000 Monte Carlo simulations were used to determined the results; and $\underline{J}_{rattle} = \bar{J}_{rattle} = 0.152\ [m]$ and $\bar{J}_{holding} = 0.034\ [m]$.

6.6 Conclusions

This section presented a new framework for the parametric optimal design of uncertain ordinary differential equation systems. The framework allows practitioners to model sources of uncertainty using the Generalized Polynomial Chaos methodology and to solve the dynamics using a least-squares collocation method. Subsequently, statistical information from the uncertain dynamics can be included in formulations of the objective function and of the constraints, to perform optimal designs under uncertainty. Robust designs benefit from the new framework when the system is nonlinear, has active constraints, or is a multi-objective optimization problem. In the case of a multi-objective optimization problem, a constraint-based formulation of the problem was shown to produce an off-set Pareto optimal trade-off surface confirming the need to directly treat uncertainties during the optimal design phase. An optimal nonlinear vehicle suspension design problem, subject to parametric uncertainty, was used to illustrate how the new framework produces an optimal design that accounts for the entire family of systems within the associated probability space. This adds robustness to the design of the optimally performing system.

7 Optimal Uncertainty Apportionment for ODE Systems

7.1 Motivation

The inclusion of uncertainty in design is of paramount practical importance because all real-life systems are affected by it. Designs that ignore uncertainty often lead to poor robustness, suboptimal performance, and higher build costs. Treatment of small geometric uncertainty in the context of manufacturing tolerances is a well studied topic. Traditional sequential design methodologies have recently been replaced by concurrent optimal design methodologies where optimal system parameters are simultaneously determined along with optimally allocated tolerances; this allows reducing manufacturing costs while increasing performance. However, the state-of-the-art approaches remain limited in that they can only treat geometric related uncertainties restricted to be small in magnitude.

This section presents a novel framework to perform robust design optimization (RDO) concurrently with optimal uncertainty apportionment (OUA) for dynamical systems governed by ordinary differential equations (ODEs). The framework considerably expands the capabilities of contemporary methods by enabling the treatment of both geometric and non-geometric uncertainties in a unified manner. Additionally, uncertainties are allowed to be large in magnitude and the governing constitutive relations may be highly nonlinear.

Uncertainties are modeled using Generalized Polynomial Chaos (gPC) and are solved quantitatively using a least-square collocation method (LSCM). The computational efficiency of this approach allows statistical moments of the uncertain system to be explicitly included in the optimization-based design process. The framework formulates design problems as constrained multi-objective optimization (cMOO) problems, thus enabling the characterization of a Pareto optimal trade-off surface that is off-set from the traditional deterministic optimal trade-off surface. The Pareto off-set is shown to be a result of the additional statistical moment information formulated in the objective and constraint relations that account for the system uncertainties. Therefore, the Pareto trade-off surface from the new framework characterizes the entire family of systems within the probability space; consequently, designers are able to produce robust and optimally performing systems at an optimal manufacturing cost.

A kinematic tolerance analysis case-study is presented first in Section 7.2 to illustrate how the new methodology can be applied to treat geometric tolerances. A nonlinear vehicle suspension design problem, subject to non-geometric parametric uncertainty, illustrates the capability of the new framework to produce an optimal design at an optimal manufacturing cost, accounting for the entire family of systems within the associated probability space. This case-study highlights the general nature of the new framework which is capable of optimally allocating uncertainties of multiple types and with large magnitudes in a single calculation.

7.2 *Tolerance Analysis* Based on gPC

To show the general nature of gPC, this section presents a gPC-based *tolerance analysis* using a one-way mechanical clutch found in a lawn mower or some other small machinery; the problem definition was borrowed from [181] and will be used for comparison purposes. The clutch assembly, shown in Figure 7.1, is comprised of an outer ring, a hub, and a roller bearing in a close-loop kinematic relation. Leveraging the system symmetry, the problem considers explicitly only a quarter of the mechanism; its independent assembly variables are $\boldsymbol{\theta} = \{a, c, d, e\}$ and dependent variables are $\boldsymbol{\lambda} = \{b, \chi, \gamma\}$**.** The basic goal of the *tolerance analysis* is to ensure that the pressure angle remains within the specified range of $6^{\circ} \leq \chi \leq 8^{\circ}$.

The assembly feature being analyzed, the pressure angle χ, is a dependent variable, therefore, there is no need for an explicit assembly feature relation as defined in (1), only the closed-loop kinematic constraint equation (2) is necessary. This two dimensional vector relation is:

$$\boldsymbol{\Phi}(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}, \boldsymbol{e}, \chi, \gamma) = \begin{cases} \vec{\boldsymbol{a}} + \vec{\boldsymbol{b}} + \vec{\boldsymbol{c}} + \vec{\boldsymbol{d}} + \vec{\boldsymbol{e}} = \boldsymbol{0} \\ \angle\boldsymbol{a} + \angle\boldsymbol{b} + \angle\boldsymbol{c} + \angle\boldsymbol{d} + \angle\boldsymbol{e} + \chi + \gamma = 0 \end{cases} \tag{121}$$

resulting in three equations for the three dependent unknowns.

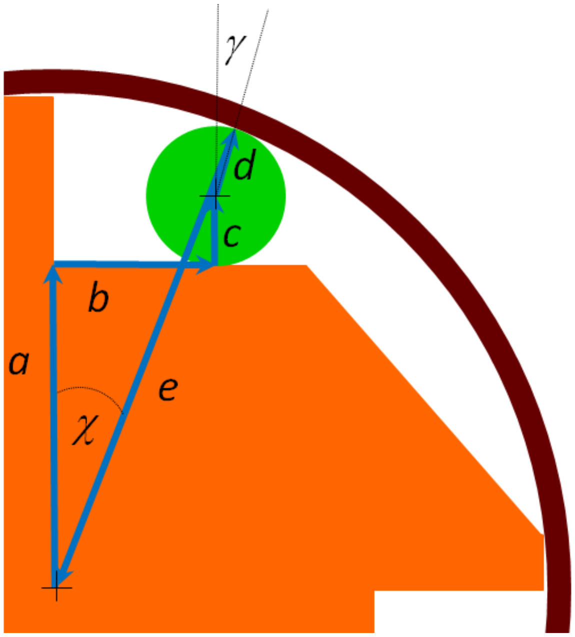

Figure 7.1—One quarter of a clutch assembly. The independent variables are dimensions $\{\boldsymbol{a}, \boldsymbol{c}, \boldsymbol{d}, \boldsymbol{e}\}$ and the dependent variables are $\{\boldsymbol{b}, \boldsymbol{\chi}, \boldsymbol{\gamma}\}$. Proper operation of the clutch requires $\mathbf{6}^{\circ} \leq \boldsymbol{\chi} \leq \mathbf{8}^{\circ}$.

The independent variables, or dimensions, are assigned tolerances where each is assumed to have a normal distribution; their respective mean and standard deviation values are presented in Table 7.1.

Table 7.1—Independent variable mean and standard deviations

Parameter	Mean (μ)	Std (σ)	Units (SI)
a	27.645	0.0167	mm
c, d	11.430	0.0033	mm
e	50.800	0.0042	mm

The analysis proceeds as presented in Section 4.2.1, where each independent variable $\theta_i(\boldsymbol{\xi})$ is approximated as shown in (11); uncertain dependent variables $\boldsymbol{\lambda}(\boldsymbol{\xi})$ are expanded using (15); and all approximations are substituted into (121) resulting in uncertain algebraic constitutive relations of the form (35). The LSCM method samples the probability space n_{cp} times and solves for the dependent variable expansion coefficients through least-squares using (29). The respective mean and standard deviation statistical moments are then efficiently determined by (32)–(33). The results of this analysis for various gPC approximation orders are shown in Table 7.2.

Table 7.2—Various results for the $3\sigma_\chi$ one-way clutch pressure angle variation

Parameter	Variation (3σ)	$\lvert\Delta_{\mathrm{MC}}\rvert$	# of samples, or collocation points	Computation time (s)
DLM	0.65788	0.00135	n/a	0.06
gPC, $p_o = 2$	0.65822	0.00101	30	0.34
gPC, $p_o = 3$	0.65900	0.00023	60	0.51
gPC, $p_o = 5$	0.65901	0.00022	168	2.73
MC	0.65923	0	2,500,000	217.42

Table 7.2 also contains the results of the analysis when applying the direct linearization method (DLM), as used in [105, 181], as well as a Monte Carlo (MC) based analysis using 2.5 million samples. Additionally, Table 7.2 reports the absolute value of the errors between the solution of the different methods when compared to the MC results, $|\Delta_{\mathrm{MC}}|$, in the third column; the number of samples used are shown in the fourth column; and the associated computation times are shown the fifth column. All methods validate that the pressure angle remains within the specified range; for example, the DLM method had the largest reported variation and its $3\sigma_\chi$ solution had a range of $6.3605^\circ \leq \chi \leq 7.6763^\circ$, which is within specification.

Using the high sample MC solution as a baseline for comparison shows that the 2nd order gPC solution results in comparable results to the DLM. The 3rd order gPC solution seems to be the best approximation point when considering both computational cost and accuracy. The reported computation times were taken from an unoptimized Mathematica code running on an HP Pavilion with the Intel i7 processor and 6 GB of RAM. Clearly the gPC approach is more computationally burdensome than the DLM, however, when considering the more general nature of the gPC approach—as was discussed in Section 4.2—and the relatively cheap cost to increase the accuracy of the solution beyond what the DLM can provide, a designer may find that this trade-off is worth the expense. In Section 7.4, the benefits of the gPC approach become more apparent when large magnitude variations and non-geometric uncertainties are included within a problem's scope.

7.3 Optimal Uncertainty Apportionment (OUA)

The new framework for simultaneous RDO and OUA of dynamical systems described by ODEs is now presented. This formulation builds upon the gPC-based uncertainty quantification techniques presented in Sections 4.2 and 4.2.2. Sources of uncertainty may come from ICs, actuator inputs, sensor outputs, and system parameters; where parametric uncertainties may include both geometric and non-geometric sources. The NLP-base formulation of the new framework is,

$$
\begin{array}{ll}
\min\limits_{\boldsymbol{x}} & \mathrm{J} = \boldsymbol{\mathcal{W}}\big(\boldsymbol{\sigma}(\boldsymbol{\xi})\big) \\
\mathrm{s.t.} & \textbf{Dynamics} \\
& \boldsymbol{\mathcal{F}}\big(\boldsymbol{q}(t;\boldsymbol{\xi}), \boldsymbol{v}(t;\boldsymbol{\xi}), \dot{\boldsymbol{v}}(t;\boldsymbol{\xi}), \boldsymbol{\theta}(t;\boldsymbol{\xi})\big) = \boldsymbol{\tau}(t;\boldsymbol{\xi}) \\
& \textbf{Kinematics} \\
& \dot{\boldsymbol{q}}(t;\boldsymbol{\xi}) = \boldsymbol{H}\big(\boldsymbol{q}(t;\boldsymbol{\xi}), \boldsymbol{\theta}(t;\boldsymbol{\xi})\big)\boldsymbol{v}(t;\boldsymbol{\xi}) \\
& \textbf{Outputs} \\
& \boldsymbol{y}(t;\boldsymbol{\xi}) = \boldsymbol{\mathcal{O}}\big(\boldsymbol{q}(t;\boldsymbol{\xi}), \dot{\boldsymbol{q}}(t;\boldsymbol{\xi}), \boldsymbol{\theta}(t;\boldsymbol{\xi})\big) \\
& \textbf{Constraints} \\
& \boldsymbol{\mathcal{C}}(\boldsymbol{y}(t;\boldsymbol{\xi}), \boldsymbol{\theta}(t;\boldsymbol{\xi}), \boldsymbol{\tau}(t;\boldsymbol{\xi}), t) \leq \mathbf{0} \\
& \textbf{Hard ICs \& TCs Conditions} \\
& \boldsymbol{q}(0;\boldsymbol{\xi}) = \boldsymbol{q}_0, \boldsymbol{q}\big(t_f;\boldsymbol{\xi}\big) = \boldsymbol{q}_{t_f} \\
& \dot{\boldsymbol{q}}(0;\boldsymbol{\xi}) = \dot{\boldsymbol{q}}_0, \dot{\boldsymbol{q}}\big(t_f;\boldsymbol{\xi}\big) = \dot{\boldsymbol{q}}_{t_f}
\end{array}
\tag{122}
$$

where the problem objective, J, is a weighted vector function, $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{n_a}$, defining n_a cost-uncertainty trade-off curves for the manufacturing cost associated with each uncertainty being apportioned. Equation (122) is subject to the dynamic constitutive relations defined in (3)–(5), and their associated ICs and optional terminal conditions (TCs). When performing simultaneous RDO and OUA, the list of optimization variables, $\boldsymbol{x} \in \mathbb{R}^{n_x}$**,** includes select nominal design parameters as well as variances, or standard deviations, of the uncertainties to be apportioned. Concurrent RDO and OUA is possible by converting the robust performance objectives of RDO to constraints and adding them to the list of problem constraints itemized in $\boldsymbol{\mathcal{C}}(\boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{\tau}, t) \leq \mathbf{0}$. The authors' work in [104] illustrates how robust performance objectives may be defined within a gPC setting. Therefore, the solution of (122) yields a system design that minimizes the manufacturing cost-to-build subject to specified robust performance criterion defined through constraints. Equation (122) is formulated as a *constrained multi-objective optimization* (cMOO) problem, meaning, as long as at least two performance constraints have opposing influences on the optimum, then a Pareto optimal set may be determined as the constraint boundaries are adjusted. For example, if each robust performance constraint, β, is bounded in the following manner $\underline{\beta} \leq \beta \leq \overline{\beta}$. A Pareto set will be obtained for unique values of $\overline{\beta}$ and/or $\underline{\beta}$ as long as the associated constraint is active. Once a given constraint becomes inactive, that constraint has no influence on the optimal value.

The NLP defined in (122) may be approached from either a *sequential nonlinear programming* (SeqNLP), or from a *simultaneous nonlinear programming* (SimNLP) perspective [65, 66]. (The literatures occasionally refers to the SeqNLP approach as *partial discretization* and to the SimNLP as *full discretization* [173].) In the SeqNLP approach, the dynamical equations (3)–(5) remain as continuous functions that may be integrated with standard off-the-shelf ODE solvers (such as Runge-Kutta). This directly leverages the LSCM-based gPC techniques described in Section 4.2 and yields a smaller optimization problem as only the optimization variables, $\boldsymbol{x}$, are discretized. On the contrary, as discussed generally in Section 3.3, the SimNLP approach discretizes (3)–(5) over the trajectory of the system and treats the complete set of equations as equality constraints for the NLP. The discretized state variables are added to $\boldsymbol{x}$ to complete the full discretization. As such, the SimNLP approach requires a slight modification in the formulation to account for the full discretization of (3)–(5) in light of the LSCM technique:

$$
\begin{aligned}
\min_{\boldsymbol{x}} \quad & \mathrm{J} = \boldsymbol{\mathcal{W}}\big(\boldsymbol{\sigma}(\boldsymbol{\xi})\big) \\
\text{s.t.} \quad & \textbf{Dynamics } ({}_1\boldsymbol{\mu}) \\
& \boldsymbol{\mathcal{F}}\left(\boldsymbol{q}({}_1\boldsymbol{\mu}), \boldsymbol{v}({}_1\boldsymbol{\mu}), \dot{\boldsymbol{v}}({}_1\boldsymbol{\mu}), \boldsymbol{\theta}({}_1\boldsymbol{\mu})\right) = \boldsymbol{\tau}({}_1\boldsymbol{\mu}) \\
& \textbf{Kinematics } ({}_1\boldsymbol{\mu}) \\
& \dot{\boldsymbol{q}}({}_1\boldsymbol{\mu}) = \boldsymbol{H}\left(\boldsymbol{q}({}_1\boldsymbol{\mu}), \boldsymbol{\theta}({}_1\boldsymbol{\mu})\right)\boldsymbol{v}({}_1\boldsymbol{\mu}) \\
& \textbf{Outputs } ({}_1\boldsymbol{\mu}) \\
& \boldsymbol{y}({}_1\boldsymbol{\mu}) = \boldsymbol{\mathcal{O}}\left(\boldsymbol{q}({}_1\boldsymbol{\mu}), \dot{\boldsymbol{q}}({}_1\boldsymbol{\mu}), \boldsymbol{\theta}({}_1\boldsymbol{\mu})\right) \\
& \textbf{Initial Conditions } ({}_1\boldsymbol{\mu}) \\
& \boldsymbol{q}(0;\ {}_1\boldsymbol{\mu}) = \boldsymbol{q}_{0,1}, \dot{\boldsymbol{q}}(0;\ {}_1\boldsymbol{\mu}) = \dot{\boldsymbol{q}}_{0,1} \\
& \qquad\qquad\qquad\qquad \vdots \\
& \textbf{Dynamics } ({}_{n_{cp}}\boldsymbol{\mu}) \\
& \boldsymbol{\mathcal{F}}\left(\boldsymbol{q}\left({}_{n_{cp}}\boldsymbol{\mu}\right), \boldsymbol{v}\left({}_{n_{cp}}\boldsymbol{\mu}\right), \dot{\boldsymbol{v}}\left({}_{n_{cp}}\boldsymbol{\mu}\right), \boldsymbol{\theta}\left({}_{n_{cp}}\boldsymbol{\mu}\right)\right) = \boldsymbol{\tau}\left({}_{n_{cp}}\boldsymbol{\mu}\right) \\
& \textbf{Kinematics } ({}_{n_{cp}}\boldsymbol{\mu}) \\
& \dot{\boldsymbol{q}}\left({}_{n_{cp}}\boldsymbol{\mu}\right) = \boldsymbol{H}\left(\boldsymbol{q}\left({}_{n_{cp}}\boldsymbol{\mu}\right), \boldsymbol{\theta}\left({}_{n_{cp}}\boldsymbol{\mu}\right)\right)\boldsymbol{v}\left({}_{n_{cp}}\boldsymbol{\mu}\right) \\
& \textbf{Outputs } ({}_{n_{cp}}\boldsymbol{\mu})
\end{aligned}
\tag{123}
$$

$$\boldsymbol{y}\left({}_{n_{cp}}\boldsymbol{\mu}\right) = \boldsymbol{\mathcal{O}}\left(\boldsymbol{q}\left({}_{n_{cp}}\boldsymbol{\mu}\right), \dot{\boldsymbol{q}}\left({}_{n_{cp}}\boldsymbol{\mu}\right), \boldsymbol{\theta}\left({}_{n_{cp}}\boldsymbol{\mu}\right)\right)$$

Initial Conditions $({}_{n_{cp}}\boldsymbol{\mu})$

$$\boldsymbol{q}\left(0; {}_{n_{cp}}\boldsymbol{\mu}\right) = \boldsymbol{q}_{0,n_{cp}}, \dot{\boldsymbol{q}}\left(0; {}_{n_{cp}}\boldsymbol{\mu}\right) = \dot{\boldsymbol{q}}_{0,n_{cp}}$$

Constraints

$$\boldsymbol{\mathcal{C}}(\boldsymbol{y}(\boldsymbol{\xi}), \boldsymbol{\theta}(\boldsymbol{\xi}), \boldsymbol{\tau}(\boldsymbol{\xi}), t) \leq \mathbf{0}$$

Equation (123) duplicates the deterministic dynamical equations (3)–(5) n_{cp} times where each set has a unique collocation point, ${}_k\boldsymbol{\mu}$. Each unique set of dynamical equations is then fully discretized and $\boldsymbol{x}$ is updated appropriately. However, the system constraints, $\boldsymbol{\mathcal{C}}(\boldsymbol{y}(\boldsymbol{\xi}), \boldsymbol{\theta}(\boldsymbol{\xi}), \boldsymbol{\tau}(\boldsymbol{\xi}), t) \leq \mathbf{0}$, are calculated using the statistical properties determined by the LSCM and the n_{cp} sets of dynamical equations. Thus, the SimNLP approach has a much larger set of constraints and optimization variables than the SeqNLP approach, but, enjoys a more structured NLP that typically experiences faster convergence.

The Direct Search (DS) class of optimization solvers—techniques such as Genetic Algorithms, Differential Evolution, and Particle Swarm—typically only treat unconstrained optimization problems. To use this kind of solver, all the inequality constraints in (122) need to be converted from *hard* constraints to *soft* constraints (as described in Section 3.4); once converted, equation (122) can be reformulated as,

$$\begin{aligned}
\min_{\boldsymbol{x}} \quad & \mathrm{J} = \boldsymbol{\mathcal{W}}\big(\boldsymbol{\sigma}(\boldsymbol{\xi})\big) \ + \mathrm{J}_{\mathrm{const}}(\mathrm{t}; \boldsymbol{\xi}) \\
\mathrm{s.t.} \quad & \textbf{Dynamics} \\
& \boldsymbol{\mathcal{F}}\big(\boldsymbol{q}(t;\boldsymbol{\xi}), \boldsymbol{v}(t;\boldsymbol{\xi}), \dot{\boldsymbol{v}}(t;\boldsymbol{\xi}), \boldsymbol{\theta}(t;\boldsymbol{\xi})\big) = \boldsymbol{\tau}(t;\boldsymbol{\xi}) \\
& \textbf{Kinematics} \\
& \dot{\boldsymbol{q}}(t;\boldsymbol{\xi}) = \boldsymbol{H}\big(\boldsymbol{q}(t;\boldsymbol{\xi}), \boldsymbol{\theta}(t;\boldsymbol{\xi})\big)\boldsymbol{v}(t;\boldsymbol{\xi}) \\
& \textbf{Outputs} \\
& \boldsymbol{y}(t;\boldsymbol{\xi}) = \boldsymbol{\mathcal{O}}\big(\boldsymbol{q}(t;\boldsymbol{\xi}), \dot{\boldsymbol{q}}(t;\boldsymbol{\xi}), \boldsymbol{\theta}(t;\boldsymbol{\xi})\big) \\
& \textbf{Initial Conditions} \\
& \boldsymbol{q}(0;\boldsymbol{\xi}) = \boldsymbol{q}_0, \dot{\boldsymbol{q}}(0;\boldsymbol{\xi}) = \dot{\boldsymbol{q}}_0
\end{aligned} \tag{124}$$

where the equality constraints from the continuous dynamics are implicit in the calculation of the objective function. This SeqNLP approach enables (124) to be solved by the DS class of unconstrained solvers.

The new framework presented in (122), (123) or (124) allows designers to directly treat the effects of modeled uncertainties during a concurrent RDO and OUA design process. The formulations are independent of the optimization solver selected; meaning, if a constrained NLP solver—such as *sequential quadratic programming* (SQP) or *interior point* (IP)—is selected, then any of the three formulations presented is appropriate, depending upon the designer's preferences regarding *hard/soft* constraint definition and partial/full discretization. On the contrary, if an unconstrained solver is selected, then (124) is the formulation of choice.

The computational efficiencies of gPC enable the inclusion of statistical moments in the OUA objective function definition as well as in the RDO constraint specifications; these statistical measures are available at a reduced computational cost as compared to contemporary techniques. However, the framework does introduce an additional layer of modeling and computation [79]. The general guidelines of when to apply the framework for RDO problems was presented in Section 6.3. From an OUA perspective, the following general guidelines can help determine if a given design will benefit from the concurrent OUA/RDO framework:

1. **Non-Geometric Uncertainties:** Traditional *tolerance allocation* techniques have been developed for the apportionment of geometric related uncertainties. The new framework provides a unified framework that enables the simultaneous apportionment of both geometric and non-geometric related uncertainties simultaneously in dynamical systems.
2. **Large Magnitude Uncertainties:** Again, traditional *tolerance allocation* techniques have been developed under the assumption that the uncertainty magnitudes are sufficiently small. This assumption is generally valid for geometric manufacturing related uncertainties, however, it may not be valid for non-geometric related uncertainties. This point is illustrated in the case study presented in Section 7.4.
3. **Simultaneous RDO/OUA Design:** As mentioned in Section 1.3.5, the research community has already found that concurrent optimal design and tolerance allocation yields a superior design than the traditional sequential optimization approach. However, the concurrent design studies to date have only treated geometric uncertainties; the new framework in (122), (123) or (124) enables RDO concurrently with OUA and treats non-geometric uncertainties in addition to the geometric.

7.4 An Illustrative Case-Study

This section illustrates the benefits of the new framework presented in Section 7.3 through a vehicle suspension design case-study where RDO and OUA are carried out concurrently. The case-study showcases OUA for a nonlinear system subject to large magnitude non-geometric uncertainties.

7.4.1 Concurrent OUA/RDO

The case-study reuses the vehicle suspension model described in Section 6.4; recall that the five uncertain system parameters are, $\boldsymbol{\theta}(\boldsymbol{\xi}) = \{m_s(\boldsymbol{\xi}), k_s(\boldsymbol{\xi}), b_s(\boldsymbol{\xi}), \eta(\boldsymbol{\xi}), k_u(\boldsymbol{\xi})\}$. The ultimate goal for an optimal vehicle suspension design is to characterize the trade-off effects between three conflicting objectives: the passenger comfort (*ride*), which is modeled as the acceleration of the sprung mass; the suspension displacement (*rattle*); and the tire road holding forces (*holding*). When applying the new concurrent OUA/RDO framework defined in Section 7.3, the optimal system performance will be determined simultaneously with the optimal manufacturing cost. This is carried out for the vehicle suspension design problem from Section 6.4 by assuming that the standard deviations of the uncertain sprung mass, $\sigma_{m_s} = m_s^1$, and tire spring constant, $\sigma_{k_u} = k_u^1$, cannot be manipulated by the design; therefore, they are treated as fixed uncertainties. It is also assumed that the mean nonlinear damping coefficient, $\mu_{\eta_s} = \eta_s^0$, is fixed. Therefore, the search variables to carry out the OUA are $\boldsymbol{x}_{OUA} = \{\, k_s^1, b_s^1, \eta_s^1\}$. The same search variables used in [104] for RDO are reused here; they are $\boldsymbol{x}_{RDO} = \{\, k_s^0, b_s^0\}$. The final list of problem search variables is the union of the two sets, or $\boldsymbol{x}_{OUA} \cup \boldsymbol{x}_{RDO}$.

There are a number of methods presented in the literature for defining a cost-uncertainty trade-off curve [107]. This case-study assumes that the *reciprocal power* cost-uncertainty trade-off curve used by the manufacturing community is a reasonable definition for the selected non-geometric uncertainties in this case study. The *reciprocal power* curve is defined as,

$$\mathcal{W}_i = A_i + \frac{B_i}{\delta_i^k}, \qquad i = 1 \ldots n_a \tag{125}$$

where A_i is a bias cost associated with the i^{th} source of uncertainty being apportioned; B_i is the cost that is scaled by the reciprocal power of the selected variation magnitude, δ_i^k, with $k \in \mathbb{R}$ defining the exponential power. Table 7.3 shows the A, B, and k values used for the uncertainties associated with $\boldsymbol{\delta}^k = \boldsymbol{x}_{OUA}$.

Table 7.3—Independent variable means and standard deviations

Parameter	Bias (A_i)	Scaled Cost (B_i)	Power (k)
k_s^1	0.5	1.40 e+6	1.5
b_s^1	0.75	3.31 e+5	2
η_s^1	0.85	7.33 e-1	1

With this selected set of search variables, defined cost-uncertainty trade-off curves, and uncertain vehicle suspension dynamics described in Section 6.4, the corresponding concurrent OUA/RDO problem may be defined as,

$$\min_{\boldsymbol{x}=\{k_s^0,k_s^1,b_s^0,b_s^1,\eta_s^1\}} \quad \sum_{i=1}^{n_a} w_i\left(A_i+\frac{B_i}{\delta_i^k}\right)$$

$$\text{s.t.}$$

Dynamics

$$\begin{aligned}
\ddot{z}_s(\xi) &= -\frac{k_s(\xi)}{m_s(\xi)}\big(z_s(\xi)-z_u(\xi)\big)-\frac{1}{m_s(\xi)}\mathcal{D}\big(\dot{z}_s(\xi),\dot{z}_u(\xi)\big)\\
\ddot{z}_u(\xi) &= \frac{k_s(\xi)}{m_u}\big(z_s(\xi)-z_u(\xi)\big)+\frac{1}{m_u}\mathcal{D}\big(\dot{z}_s(\xi),\dot{z}_u(\xi)\big)\\
&\qquad -\frac{k_u(\xi)}{m_u}\big(z_u(\xi)-z_g\big)
\end{aligned}$$

Outputs

$$\boldsymbol{y}=\begin{bmatrix} z_s(\xi)\\ \dot{z}_s(\xi)\\ z_u(\xi)\\ \dot{z}_u(\xi)\\ \int_0^{\mathrm{t_f}}\big(\ddot{z}_s(\xi)\big)^2\,dt\\ rms\big(z_u(\xi)-z_g\big)\end{bmatrix} \tag{126}$$

Constraints

$$\begin{aligned}
&J_{ride}(\xi)-\overline{J}_{ride}\le 0\\
&J_{rattle}(\xi)-\overline{J}_{rattle}\le 0,\ \ (extension)\\
&\underline{J}_{rattle}-J_{rattle}(\xi)\le 0,\ \ (compression)\\
&J_{holding}(\xi)-\overline{J}_{holding}\le 0\\
&\underline{k}_s\le k_s^0(\xi)\le \overline{k}_s\\
&\underline{b}_s\le b_s^0(\xi)\le \overline{b}_s
\end{aligned}$$

Initial Conditions

$$\boldsymbol{z}(0)=\boldsymbol{z}_0,\ \dot{\boldsymbol{z}}(0)=\dot{\boldsymbol{z}}_0$$

where w_i is a scalarization weighting factor for the i^{th} apportionment cost and the uncertain asymmetric damping force is,

$$\mathcal{D}\big(\dot{z}_s(\xi), \dot{z}_u(\xi)\big) = \begin{cases} b_s(\xi)\big(\dot{z}_s(\xi) - \dot{z}_u(\xi)\big), & \big(\dot{z}_s(\xi) - \dot{z}_u(\xi)\big) \geq 0 \\ \eta(\xi)\, b_s(\xi)\big(\dot{z}_s(\xi) - \dot{z}_u(\xi)\big), & \big(\dot{z}_s(\xi) - \dot{z}_u(\xi)\big) \geq 0 \end{cases} \tag{127}$$

Therefore, (126) simultaneously performs RDO and OUA subject to the uncertain system dynamics defined in Section 6.4 and opposing performance constraints for vehicle *ride*, *rattle*, and *holding*. In other words, (126) determines the optimal apportionment of the uncertainties in $\boldsymbol{x}_{OUA}$ that satisfy the performance constraints; this is accomplished by simultaneously determining optimal nominal suspension values in $\boldsymbol{x}_{RDO}$. Equation (126) is a robust cMOO formulation in that a Pareto optimal set accounting for the system's uncertainties may be constructed by sweeping through a range of the performance constraint bounds, $\left\{\bar{J}_{ride}, \underline{J}_{rattle}, \bar{J}_{rattle}, \bar{J}_{holding}\right\}$. Recalling the mean and standard deviation definitions provided in (32)–(33), the performance constraint definitions are,

$$\begin{gathered} \left(\mu_{y_5} + a_1\sigma_{y_5}\right) - \bar{J}_{ride} \leq 0 \\ \Big(\left(\mu_{z_s} + a_2\sigma_{z_s}\right) - \left(\mu_{z_u} - a_3\sigma_{z_u}\right)\Big) - \bar{J}_{rattle} \leq 0 \\ \underline{J}_{rattle} - \Big(\left(\mu_{z_s} - a_4\sigma_{z_s}\right) - \left(\mu_{z_u} + a_5\sigma_{z_u}\right)\Big) \leq 0 \\ \left(\mu_{y_6} + a_6\sigma_{y_6}\right) - \bar{J}_{holding} \leq 0 \end{gathered} \tag{128}$$

Therefore, the performance constraints bound the mean values plus or minus a weighted standard deviation. The constants a_i are weighting factors of the standard deviations; by setting $a_i > 1.0$ yields a more conservative design. The interested reader is referred to [104] for more details regarding the definition of the uncertain performance constraints and this approach to robust cMOO.

7.4.2 Results

This case study also reused with the same PDF approximation order, number of basis terms and collocation points as used in Section 6.5.

The results show the progression of the suspension design from a deterministic optimal performance design, to an RDO design, and finally the concurrent OUA/RDO design based on the new framework. Figure 7.2 best helps visualize the differences between the solutions obtained from the three designs; in the following figures the deterministic design optimization is referenced as *dOpt*; RDO is referenced as *uOpt*; and the simultaneous OUA/RDA is referenced as *aOpt*.

Figure 7.2 presents a specific 3D optimal solution in the constraint space as projections onto the three orthogonal 2D planes, $holding - rattle, holding - ride, rattle - ride$; this solution is obtained when the bounding constraints are set to $\bar{J}_{ride} = 220\ [m^2/s^3]$, $\underline{J}_{rattle} = \bar{J}_{rattle} = 0.203\ [m]$, and $\bar{J}_{holding} = 0.034\ [m]$. The *dOpt* solution is represented by a solid dot and results in an active *holding* constraint. The mean of the *uOpt* solution is represented by an asterisk. When all weighting factors are set to $a_i = 1$, the mean solution is enclosed in a 3D cuboid whose dimensions are determined by the original non-optimal uncertainty standard deviations. The projections of the *uOpt* cuboid are denoted by the dotted lines. Upon close inspection of Figure 7.2 it is apparent that the *uOpt* solution has an active *rattle* constraint. Since the σ_{rattle}^{uOpt} is so large, the *uOpt* solution was pushed to a significantly lower $\mu_{holding}^{uOpt}$ value when compared to the deterministic $x_{holding}^{dOpt}$ value.

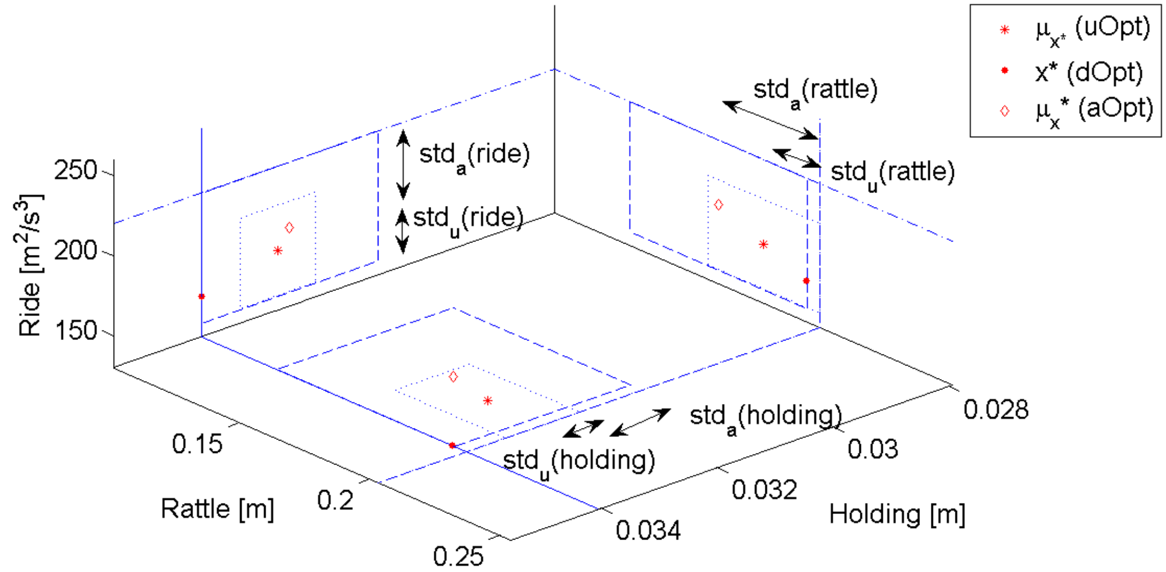

Figure 7.2—Projection of the 3D *dOpt*, *uOpt* , and *aOpt* optimal solutions onto the three orthogonal 2D planes. These optimal solutions were determined when: $\bar{J}_{ride} = 220\ [m^2/s^3]$, $\underline{J}_{rattle} = \bar{J}_{rattle} = 0.203\ [m]$, and $\bar{J}_{holding} = 0.034\ [m]$. Notice how each solution has a different set of active constraints.

Finally, the optimal mean solution obtained from the new framework for a simultaneous OUA/RDO design is represented by a diamond. Again, when all weighting factors are set to $a_i = 1$, the mean solution is enclosed in a 3D cuboid whose dimensions are determined by the optimally apportioned uncertainty standard deviations, $\boldsymbol{x}_{OUA}$. The projections of the *aOpt* cuboid are denoted by the dashed lines. Notice how the apportioned uncertainties for the *aOpt* solution are significantly larger than the *uOpt* solution. Also, the result of the uncertainty apportionment yields active *ride* and *holding* constraints. Figure 7.2 confirms that $\boldsymbol{x}_{OUA}$ and $\boldsymbol{x}_{RDO}$ are coupled in that the *aOpt* solution is shifted when compared to that found by *uOpt*; therefore, the simultaneous search finds the true optimal solution.

A Pareto optimal trade-off surface may be obtained by sweeping through a range of values of the bounding constraints. Since there are four performance constraints the actual Pareto trade-off is a 5D surface; however, to illustrate the *ride* and *holding* constraints are fixed and the OUA/RDO optimal solution is determined for a range of values of the *rattle* bound; where $\underline{J}_{rattle} = \bar{J}_{rattle}$. The resulting Pareto surface may be illustrated as a 2D projection into the objective/*rattle* plane; this is illustrated in Figure 7.3.

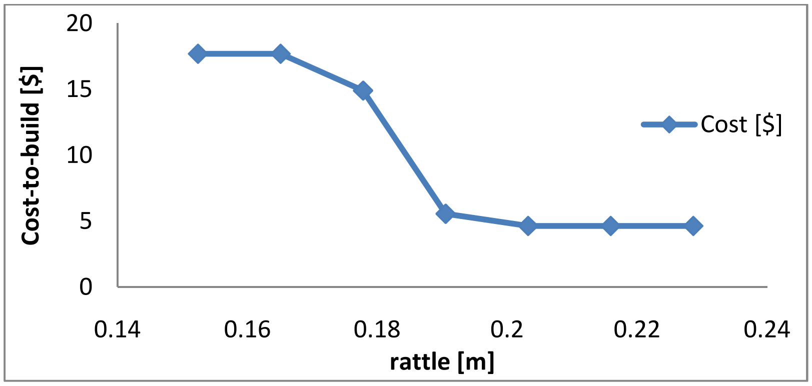

Figure 7.3—A single 2D plane from the 5D Pareto optimal set showing the trade-off between the cost-to-build objective and *rattle* constraint bound; the other constraints are held constant at $\bar{J}_{ride} = 220\ [m^2/s^3]$, and $\bar{J}_{holding} = 0.034\ [m]$. The first two points on the curve are set to zero as these resulted in infeasible *aOpt* designs.

One interesting observation is that the OUA/RDO framework determined the first two design points shown in Figure 7.3 to be infeasible points; therefore, their values were set to the objective value that results if no optimization was performed. The reason for these infeasible constraints is best illustrated by Figure 7.4, which shows the 2D projection of the *uOpt ride* objective trade-off with the *rattle* constraint resulting from an RDO, or *uOpt*, design [104]; the *holding* bound is fixed at $\bar{J}_{holding} = 0.034\ [m]$. The diamond curve is obtained from a *dOpt* optimal design and the square curve is obtained from the RDO design, however, the triangle line shows the bounding constraint value assigned for the OUA/RDO, or *aOpt*, design ($\bar{J}_{ride} = 220\ [m^2/s^3]$). When the *ride* bound (triangle line) is below the *uOpt* curve (square curve) the OUA/RDO design requires the uncertainties to be reduced, or tightened, to satisfy the constraints. However, since the variations of the uncertain mass, m_s^1, and tire spring constant, k_u^1, were assumed to be uncontrollable, the required uncertainty reductions for the first two points ended up being infeasible.

The resulting optimally allocated uncertainties for the various design points illustrated in Figure 7.3 are shown in Figure 7.5. The optimal distribution of the allocated uncertainties is a function of the selected cost-uncertainty trade-off curves defined by (125) and each associated weighting factor w_i; Table 7.4 documents the apportionment results obtained from the specific case when $\bar{J}_{ride} = 220\ [m^2/s^3]$, $\underline{J}_{rattle} = \bar{J}_{rattle} = 0.203\ [m]$, and $\bar{J}_{holding} = 0.034\ [m]$. The final apportionment of optimal standard deviations resulted in changes as large as 300% relative to their respective initial values and as large as 64% relative to their respective mean values; this clearly show-cases the new concurrent OUA/RDO framework's ability to treat uncertainties with large magnitudes.

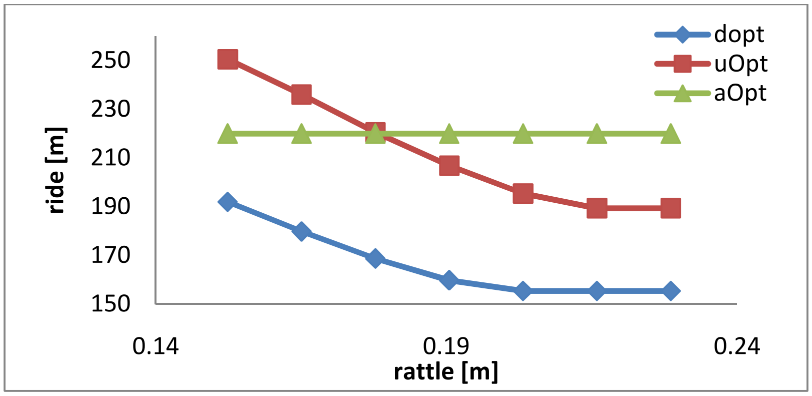

Figure 7.4—A single 2D plane showing the trade-off between the *ride* objective and *rattle* constraint bound when performing an RDO design; where $\bar{J}_{holding} = 0.034\ [m]$ is held fixed. The triangle curve represents the bounding value for $\bar{J}_{ride}$ when performing the OUA/RDO designs associated with Figure 7.2 and Figure 7.3. The first two points on the triangle curve require too large of a reduction in uncertainties and therefore result in infeasible *aOpt* designs.

The final resulting cost-to-manufacture for the *dOpt* and *uOpt* designs was $17.70; this is based on the initial non-optimal uncertainty levels which apply to both designs. However, the *aOpt* solutions result in significant cost-to-manufacture reductions; Figure 7.3 shows the cost savings for the various design points, where savings as high as 74% were experienced. Clearly, the actual cost-savings achievable by applying the new simultaneous OUA/RDO framework defined in (122), (123) or (124) is very dependent upon the definition of the respective cost-uncertainty trade-off curves, however, this case-study illustrates the power of treating uncertainty up-front during the design process; robust optimally performing systems are designed at an optimal cost-to-manufacture.

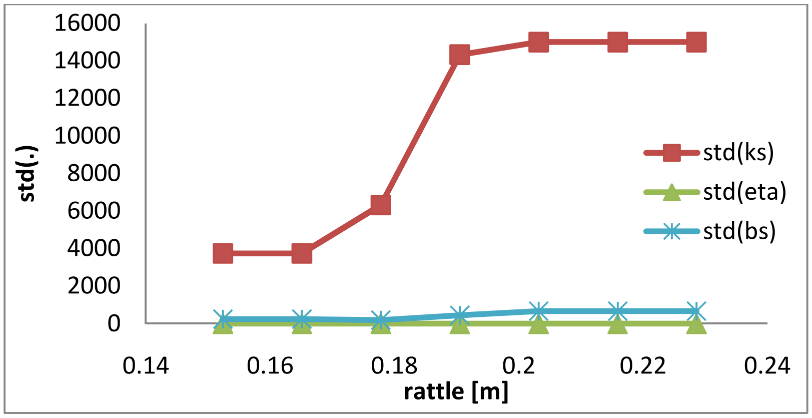

Figure 7.5—Optimally apportioned uncertainties determined from the new simultaneous OUA/RDO framework. The first two points were infeasible designs and therefore the values were set to their initial values.

Table 7.4—Apportionment results obtained from the new concurrent OUA/RDO framework corresponding to the case when $\bar{J}_{ride} = 220\ [m^2/s^3]$, $\underline{J}_{rattle} = \bar{J}_{rattle} = 0.203\ [m]$, and $\bar{J}_{holding} = 0.034\ [m]$.

	Δ from non-optimal	% Δ from non-optimal	% from μ
k_s^1	11250	300.00%	64.27%
b_s^1	423.0763	20.19%	29.70%
η_s^1	0.5213	300.12%	50.00%

7.4.3 Conclusions

This section presented a novel framework that enables RDO concurrently with OUA for dynamical systems described by ODEs. Current state-of-the-art methodologies are limited to only treating geometric uncertainties of small magnitude. The new framework removes these limitations by treating both geometric and non-geometric uncertainties in a unified manner within a concurrent RDO/OUA setting. Additionally, uncertainties may be relatively large in magnitude and the system constitutive relations may be highly nonlinear. The vehicle suspension design case-study supports this message; uncertainty allocations on the order of 300% of the initial value were obtained.

The computational efficiency of the selected gPC approach allows statistical moments of the uncertain system to be explicitly included in the optimization-based design process. The framework, presented in a cMOO formulation, enables a Pareto optimal trade-off surface characterization for the entire family of systems within the probability space. The Pareto trade-off surface from the new framework is shown to be off-set from the traditional deterministic optimal trade-off surface as a result of the additional statistical moment information formulated into the objective and constraint relations. As such, the vehicle suspension case-study Pareto trade-off surface from the new framework enables a more robust and optimally performing design at an optimal manufacturing cost.

8 Conclusions and Future Research Directions

This section provides a summary of the research presented previously as well as several directions that should be considered for future work.

8.1 Conclusions

This research focuses on improving the state-of-the-art in computational optimization-based design approaches. This focus is motivated by the observation that all real-life systems are subject to uncertainties and that their presence reduces system robustness and performance, as well as increases manufacturing costs. Additionally, contemporary design optimization methods are limited in their ability to efficiently account for the adverse effects of uncertainty during the design process. Some of these limitations are:

1. They are prohibitively inefficient in accounting for uncertainty in general nonlinear systems.
2. They are limited to specific types of uncertainty (e.g., sensor/actuator noise) and distributions (e.g., only Gaussian).
3. They do not leverage the actuation scheme of controlled dynamical systems to gain computation efficiencies in an optimal control, or motion planning, setting.
4. They do not enable the optimal apportionment of uncertainty concurrently with robust optimal design for non-geometric related uncertainties.

This research successfully removes these limitations by developing a new computational framework, based upon NLP and gPC/LSCM, that efficiently accounts for uncertainties during a parametric optimal design process. All sources of uncertainties quantified with gPC are treated in a unified manner. Also, computation of the uncertain dynamics' statistical moments with LSCM is significantly faster than traditional methods. This efficiency makes the treatment of uncertainty during dynamic optimization feasible; where statistical moments of the uncertain dynamics are including in the NLP's constraints and objective function. Robust constraint satisfaction is also scalable within this framework where standard deviations may be arbitrarily scaled to obtain a desired balance between performance and constraint satisfaction across the probability space.

The framework developed is general in that it is independent of the methodology used to develop the dynamical EOMs; both simultaneous and recursive dynamics formulations are directly applicable when using LSCM. Additionally, since the framework is formulated as a NLP, standard solvers such as SQP, IP, and Differential Evolution are directly applicable.

The impact of the new framework is significant in that design engineers are now able to directly treat system uncertainties during the dynamic optimization process efficiently; this enables designers, for the first time, to answer design questions that are applicable to the entire family of realizable systems in the probability space. Ultimately, the framework aids engineers in designing robust optimally performing systems at an optimal manufacturing cost.

The framework was shown to successfully enable the efficient design of robust optimally performing systems through the following three application areas:

- **Motion planning of Uncertain Systems:** Additional efficiencies in the framework are gained by leveraging the actuation scheme of dynamical systems. This was accomplished by developing the framework for forward, inverse, and hybrid dynamics formulations; thus accounting for both fully-actuated and under-actuated systems. Also, specific motion planning objective functions and constraints were defined that leverage the computational efficiencies of the computed uncertainty moments. Ultimately, designers are now able to determine optimally performing motion plans that robustly satisfy system constraints. The following publications resulted from this specific development effort [78-81].
- **Parametric robust design optimization (RDO):** Direct treatment of system uncertainties during RDO enables designers, for the first time, to characterize the Pareto optimal trade-off surface for the entire family of realizable systems; this yields a shifted, or off-set, Pareto optimal trade-off surface—when compared to contemporary deterministic RDO approaches—that more accurately characterizes the optimal designs for the entire probability space. Problems that are nonlinear, have active constraints, or opposing design objectives, were shown to benefit from this new framework. Also, the new framework used a constraint-based multi-objective optimization (cMOO) formulation which revealed the relationship between the magnitude of the Pareto optimal surface shift and the scaled standard deviations in the constraint definitions. Again, designers are now able to determine optimally performing designs that are configurably robust across the entire family of realizable systems. The following publications resulted from this specific development effort [103, 104].
- **Optimal uncertainty apportionment (OUA):** The first time, design engineers are able to treat both geometric and non-geometric uncertainties of relatively large magnitude during a concurrent RDO/OUA design process. Performance objectives are redefined as

specified bounding constraints yielding a cMOO formulation where the new problem objective is a weighted function of the various manufacturing cost-uncertainty trade-off curves. Subsequently, designers are now able to determine optimal manufacturing costs for a robust system design that accomplishes a specified performance level. The following publication resulted from this specific development effort [114].

8.2 Future Research Directions

The frameworks developed in this research effort primarily focus on dynamical systems described by ODEs. Many engineered systems require more complex models to capture significant dynamical behaviors. For example, dynamical systems that include kinematic closed-loops frequently are very difficult, or impossible, to model by an ODE, but may be accurately modeled with DAEs. Additionally, both ODEs and DAEs describe lumped parameter systems; in order to model distributed parameter systems, such a flexible multibody systems, PDEs are typically used. Therefore, it is recommended that future investigations consider building upon the framework developed in this research to provide robust optimal design capabilities for dynamical systems described by uncertain DAEs, and PDEs.

Additionally, the new framework makes use of gPC to efficiently quantify the uncertainties in the dynamical systems. As previously mentioned, gPC has a C^1 continuity assumption. Therefore, another recommendation for future research is to investigate optimal design of discontinuous uncertain dynamical systems. One practical example of discontinuous dynamical systems is those constrained by unilateral constraints, such as geometric contact; these systems are frequently modeled with DAIs. The impulsive forces experienced when the constraint is at equality result in velocities that are not C^1 continuous. Therefore, systems with impulsive geometric contact forces cannot benefit from the new framework and require additional research to enable efficient robust design optimization.

Finally, the applicability of the new framework to chaotic dynamical systems—such as, strange attractor systems or systems that experience bifurcations—was not within the scope of these investigations. It is anticipated that the framework developed in this work will need modification in order to properly treat chaotic systems, therefore, it is recommended for future investigations.

Appendix A: Derivation of Uncertain Objective and Constraint Terms

This appendix derives the computational aspects of the gPC-based moments and custom terms presented for motion planning in Section 5.4. First, the basic moments will be derived followed by those for motion planning

8.3 Mean, Variance and Standard Deviation

In Section 4.2 it was shown that any C^1 continuous source of uncertainty, θ, can be approximated by an orthogonal polynomial series expansion with a finite number of terms, p_o; this expansion takes the form of,

$$\theta(\xi) = \sum_{j=0}^{p_o} \theta^j \psi^j \left(\xi(\omega)\right) \tag{129}$$

here $\theta^j \in \mathbb{R}$ are known coefficients; $\psi^j \in \mathbb{R}$ are individual single dimensional orthogonal polynomial bases; $\xi(\omega) \in \mathbb{R}$ is a random variable that maps the random event $\omega \in \Omega$, from the sample space, Ω, to the domain of the orthogonal polynomial basis (e.g., $\xi\colon \Omega \to [-1,1]$). It was also explained that gPC provides a variety of polynomial-distribution pairs that can be used as the bases in (129). Recalling the definition of the mean for a continuous random variable x,

$$\mu_\theta = E[\theta(\xi)] = \int_{\boldsymbol{\Omega}} \theta(\xi)\, f(\xi) d\xi \tag{130}$$

where $f(\xi)$ is an appropriately selected weighting function.

If, for example, a Beta distribution is selected to model the uncertain variable θ, then Jacobi polynomials are used in the expansion (129) and the weighting function $f(\xi)$ becomes the joint probability density function of the Beta distribution. Substitute (129) into (130) yields,

$$\mu_\theta = E[\theta(\xi)] = \int_{-1}^{1} \sum_{j=0}^{p_o} \theta^j \psi^j \left(\xi\right) f(\xi) d\xi \tag{131}$$

where the random event ω has been dropped for notational brevity, and the domain is appropriately updated to $\xi : \Omega \to [-1,1]$. It turns out that for many of the distributions (e.g., Normal, Beta, and Uniform) the following properties are true,

$$\int_{\Omega} \psi^0(\xi) f(\xi) d\xi = constant \tag{132}$$

and,

$$\int_{\Omega} \psi^j(\xi)\, f(\xi) d\xi = 0, \qquad j = 1 \dots p_o. \tag{133}$$

Therefore, the mean of a gPC approximated random variable is simply,

$$\mu_\theta = E[\theta(\mathrm{t}; \xi)] = \theta^0 \int_{\Omega} \psi^0(\xi) f(\xi) d\xi \tag{134}$$

And the variance is,

$$\sigma_\theta^2 = Var[\theta(\xi)] = \int_{\Omega} (\theta(\xi) - \mu_\theta)^2\, f(\xi) d\xi = E[(\theta(\xi) - \mu_\theta)^2] \tag{135}$$

Recalling the orthogonality property of these polynomials,

$$\langle \psi^i(\xi), \psi^j(\xi) \rangle \; = \; \int_{\Omega} \psi^i(\xi) \psi^j(\xi) f(\xi) d\xi \; = \; 0, \quad \text{for i≠j} \tag{136}$$

Therefore, substituting (129) into (135) and applying (136) yields,

$$\begin{aligned} \sigma_\theta^2 &= \int_{\Omega} \left(\sum_{j=1}^{p_o} \theta^j \psi^j\,(\xi) \right)^2 f(\xi) d\xi \\ &= \sum_{j=1}^{p_o} \left(\left(\theta^j\right)^2 \int_{\Omega} \left(\psi^j(\xi)\right)^2 f(\xi) d\xi \right) \\ &= \sum_{j=1}^{p_o} \left(\left(\theta^j\right)^2 \langle \psi^i(\xi), \psi^j(\xi) \rangle \right) \end{aligned} \tag{137}$$

Recall that $\langle \psi^i(\xi), \psi^j(\xi) \rangle = 1, \forall j$ when using *normalized basis*; *standardized basis* are constant and may be computed off-line for efficiency using (136). Therefore, if a *normalized basis* then

$\mu_\theta = \theta^0$ and the variance is $\sigma_\theta^2 = \sum_{j=1}^{p_o} \left(\theta^j\right)^2$. Also, the standard deviation, $\sigma_\theta = \sqrt{Var[\theta(\xi)]} = \sqrt{\sum_{j=1}^{p_o} (\theta^j)^2}$. As a result, the moment computations for a random variable modeled with gPC reduce to a simple set of efficient arithmetic operations.

This presentation has been made using single dimensional basis created from orthogonal polynomials; however, it's important to note that all of these properties hold for multi-dimensional basis created from the contributing single dimensional basis.

8.4 Derivation of the Computational Approach for the Motion Planning Uncertainty Objective Terms

In Section 5.4, uncertain objective terms were presented for the uncertain motion planning framework. Additional details will now be presented to show the derivation of the efficient computational approaches provided in Section 5.4.

8.4.1 Expected Terminal Output Error

Applying the computational reductions for the mean presented in Section 8.3, the expected error between a reference and an uncertain variable can be derived as,

$$\begin{aligned} E\left[e\left(t_f;\xi\right)\right] &= E\left[y_{ref}\left(t_f\right) - y\left(t_f;\xi\right)\right] \\ &= y_{ref}\left(t_f\right) - E\left[y\left(t_f;\xi\right)\right] \\ &= y_{ref}\left(t_f\right) - y^0(t_f)\langle\psi^0,\psi^0\rangle \end{aligned} \tag{138}$$

which corresponds to the scalar version of (85) presented in Section 5.4.

8.4.2 Uncertain Terminal Output Variance

Applying the computational reductions for the variance presented in Section 8.3, the error variance between a reference and an uncertain variable can be derived as,

$$\begin{aligned} Var[e(\mathrm{t_f};\xi)] &= E\left[\left(e(\mathrm{t_f};\xi) - \mu_{e(t_f)}\right)^2\right] \\ &= E\left[\left(\left(y_{ref}\left(t_f\right) - y\left(t_f;\xi\right)\right) - \left(y_{ref}\left(t_f\right) - y^0(t_f)\langle\psi^0,\psi^0\rangle\right)\right)^2\right] \\ &= E\left[\left(y^0(t_f)\langle\psi^0,\psi^0\rangle - y\left(t_f;\xi\right)\right)^2\right] \end{aligned} \tag{139}$$

$$
\begin{aligned}
&= E\left[\left(y^0(t_f)\langle\psi^0,\psi^0\rangle\right)^2 - 2y^0\left(t_f\right)\langle\psi^0,\psi^0\rangle y\left(t_f;\xi\right) + \left(y\left(t_f;\xi\right)\right)^2\right] \\
&= E\left[\left(y^0(t_f)\langle\psi^0,\psi^0\rangle\right)^2\right] - E\left[2\, y^0\left(t_f\right)\langle\psi^0,\psi^0\rangle\, y\left(t_f;\xi\right)\right] \\
&\qquad\qquad + E\left[\left(y\left(t_f;\xi\right)\right)^2\right] \\
&= 2\left(y^0(t_f)\langle\psi^0,\psi^0\rangle\right)^2 - 2\left(y^0(t_f)\langle\psi^0,\psi^0\rangle\right)^2 + E\left[\left(y\left(t_f;\xi\right)\right)^2\right] \\
&= \sum_{j=0}^{n_b}\left(y^j(\mathrm{t_f})\right)^2\langle\psi^j,\psi^j\rangle
\end{aligned}
$$

which corresponds to the scalar version of (86) presented in Section 5.4.

8.4.3 Expected Input Effort

Applying the definition of the mean in (130) and making use of the orthogonality principle in (136), the uncertain input effort objective penalty term can be derived as,

$$
\begin{aligned}
E[\tau(t;\xi)^2] &= E\left[\left(\textstyle\sum_{j=0}^{p_o}\tau^j(t)\psi^j(\xi)\right)^2\right] \\
&= \int_{\boldsymbol{\Omega}}\left(\sum_{j=0}^{p_o}\tau^j(t)\psi^j(\xi)\right)^2 f(\xi)d\xi \\
&= \sum_{j=0}^{p_o}\left(\tau^j(t)\right)^2\langle\psi^j,\psi^j\rangle
\end{aligned}
\tag{140}
$$

Equation (140) is a scalar version of the expected effort computations; when (140) is integrated over the trajectory of the system the resulting quantity corresponds to (82).

8.4.4 Expected Input Power

The expected power counter-part to (140), when both the input torque and output rate are uncertain, can be derived as,

$$
\begin{aligned}
E[\tau(t;\xi)\mathrm{y}(t;\xi)] &= E\left[\sum_{j=0}^{p_o}\tau^j(t)\psi^j(\xi)\sum_{j=0}^{p_o}\mathrm{y}^j(t)\psi^j(\xi)\right] \\
&= \sum_{j=0}^{p_o}\tau^j(t)\,\mathrm{y}^j(t)\,\langle\psi^j,\psi^j\rangle
\end{aligned}
\tag{141}
$$

where $\mathrm{y}(\boldsymbol{\xi}, t) = \dot{q}(\boldsymbol{\xi}, t)$. Again, equation (141) is a scalar version of the expected power computations; when (141) is integrated over the trajectory of the system the resulting quantity corresponds to (83).

In the event that only the input torque or the output rate is uncertain, then the corresponding expected power can be derived by,

$$\begin{aligned} E[\tau(t)\mathrm{y}(t;\xi)] &= \tau(t)\, E\left[\sum_{j=0}^{p_o} \mathrm{y}^j(t)\psi^j(\xi)\right] \\ &= \tau(t)\, \mathrm{y}^0(t)\, \langle \psi^0, \psi^0 \rangle, \text{ or,} \\ &= \mathrm{y}(t)\, \tau^0(t)\, \langle \psi^0, \psi^0 \rangle \end{aligned} \tag{142}$$

8.4.5 Expected Input Jerk

The derivation of the expected input jerk is analogous to that of the expected torque,

$$\begin{aligned} E[\dot{\tau}(t;\xi)^2] &= E\left[\left(\sum_{j=0}^{p_o} \dot{\tau}^j(t)\psi^j(\xi)\right)^2\right] \\ &= \int_{\boldsymbol{\Omega}} \left(\sum_{j=0}^{p_o} \dot{\tau}^j(t)\psi^j(\xi)\right)^2 f(\xi)d\xi \\ &= \sum_{j=0}^{p_o} \left(\dot{\tau}^j(t)\right)^2 \langle \psi^j, \psi^j \rangle \end{aligned} \tag{143}$$

Equation (143) is a scalar version of the expected effort computations; when (143) is integrated over the trajectory of the system the resulting quantity corresponds to (84).

References

[1] Papoulis, A., Pillai, S., and Unnikrishna, S., 2002, *Probability, Random Variables, and Stochastic Processes*, McGraw-Hill, New York.

[2] Rensburg, E., and Torrie, G., 1993, "Estimation of Multidimensional Integrals: Is Monte Carlo the Best Method?," Journal of Physics A: Mathematical and General, 26, pp. 943.

[3] Bratley, P., Fox, B., and Niederreiter, H., 1992, "Implementation and Tests of Low-Discrepancy Sequences," ACM Transactions on Modeling and Computer Simulation (TOMACS), 2 (3), pp. 213.

[4] Wiener, N., 1938, "The Homogeneous Chaos," American Journal of Mathematics, 60 (4), pp. 897-936.

[5] Xiu, D., and Karniadakis, G., 2002, "The Wiener-Askey Polynomial Chaos for Stochastic Differential Equations," SIAM Journal of Scientific Computing, 24 (2), pp. 619-644.

[6] Xiu, D., 2009, "Fast Numerical Methods for Stochastic Computations: A Review," Communications in Computational Physics, 5 (2-4), pp. 242-272.

[7] Xiu, D., and Hesthaven, J. S., 2005, "High-Order Collocation Methods for Differential Equations with Random Inputs," SIAM Journal on Scientific Computing, 27 (3), pp. 1118-1139.

[8] Xiu, D., 2007, "Efficient Collocational Approach for Parametric Uncertainty Analysis," Communications in Computational Physics, 2 (2), pp. 293-309.

[9] Sandu, A., Sandu, C., and Ahmadian, M., 2006, "Modeling Multibody Systems with Uncertainties. Part I: Theoretical and Computational Aspects," Multibody System Dynamics, 15 (4), pp. 369-391.

[10] Cheng, H., and Sandu, A., 2009, "Efficient Uncertainty Quantification with the Polynomial Chaos Method for Stiff Systems," Mathematics and Computers in Simulation, 79 (11), pp. 3278-3295.

[11] Wan, X., and Karniadakis, G., 2005, "An Adaptive Multi-Element Generalized Polynomial Chaos Method for Stochastic Differential Equations," Journal of Computational Physics, 209 (2), pp. 617-642.

[12] Wan, X., and Karniadakis, G., "Adaptive Numerical Solutions of Stochastic Differential Equations"

[13] Wan, X., and Karniadakis, G., 2006, "Beyond Wiener–Askey Expansions: Handling Arbitrary Pdfs," Journal of Scientific Computing, 27 (1), pp. 455-464.

[14] Wan, X., and Karniadakis, G., 2007, "Multi-Element Generalized Polynomial Chaos for Arbitrary Probability Measures," SIAM Journal on Scientific Computing, 28 (3), pp. 901-928.

[15] Foo, J., Wan, X., and Karniadakis, G., 2008, "The Multi-Element Probabilistic Collocation Method: Error Analysis and Simulation," J. Comput. Phys, 227 (22), pp. 9572–9595.

[16] Foo, J., and Karniadakis, G. E., 2010, "Multi-Element Probabilistic Collocation Method in High Dimensions," Journal of Computational Physics, 229 (5), pp. 1536-1557.

[17] Sandu, C., Sandu, A., and Ahmadian, M., 2006, "Modeling Multibody Systems with Uncertainties. Part Ii: Numerical Applications," Multibody System Dynamics, 15 (3), pp. 241-262.

[18] Blanchard, E., Sandu, A., and Sandu, C., 2007, "Parameter Estimation Method Using an Extended Kalman Filter," Joint North America, Asia-Pacific ISTVS Conference, Fairbanks, Alaska, USA, pp. 23-26.

[19] Blanchard, E., Sandu, C., and Sandu, A., 2007, "A Polynomial-Chaos-Based Bayesian Approach for Estimating Uncertain Parameters of Mechanical Systems," Proceedings of the ASME IDETC 2007, 9th International Conference on Advanced Vehicle and Tire Technology, Las Vegas, NV, USA, pp. 4-7.

[20] Shimp Iii, S., 2008, "Vehicle Sprung Mass Parameter Estimation Using an Adaptive Polynomial-Chaos Method," Masters thesis, Virginia Tech, Blacksburg.

[21] Southward, S., 2007, "Real-Time Parameter Id Using Polynomial Chaos Expansions," ASME Conference Proceedings, 2007 (43033), pp. 1167-1173.

[22] Pence, B. L., Fathy, H. K., and Stein, J. L., 2009, "A Base-Excitation Approach to Polynomial Chaos-Based Estimation of Sprung Mass for Off-Road Vehicles," ASME Dynamic Systems and Control Conference, n PART A, pp. 857-864.

[23] Pence, B. L., Fathy, H. K., and Stein, J. L., 2010, "Recursive Bayesian Parameter Estimation Using Polynomial Chaos Theory"

[24] Pence, B. L., Fathy, H. K., and Stein, J. L., 2010, "An Integrated Cost/Maximum Likelihood Approach to Recursive Polynomial Chaos Parameter Estimation"

[25] Blanchard, E., 2010, "Polynomial Chaos Approaches to Parameter Estimation and Control Design for Mechanical Systems with Uncertain Parameters," Doctorate thesis, Virginia Tech, Blacksburg.

[26] Blanchard, E., Sandu, C., and Sandu, A., 2009, "Comparison between a Polynomial-Chaos-Based Bayesian Approach and a Polynomial-Chaos-Based Ekf Approach for Parameter Estimation with Application to Vehicle Dynamics," Proceedings of the ASME 2009 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, 11th International Conference on Advanced Vehicle and Tire Technology, San Diego, CA, USA

[27] Blanchard, E., Sandu, A., and Sandu, C., 2010, "Polynomial Chaos-Based Parameter Estimation Methods Applied to a Vehicle System," Proceedings of the Institution of Mechanical Engineers, Part K: Journal of Multi-body Dynamics, 224 (1), pp. 59-81.

[28] Blanchard, E., Sandu, A., and Sandu, C., 2009, "Parameter Estimation for Mechanical Systems Via an Explicit Representation of Uncertainty," Engineering Computations, 26 (5), pp. 541-569.

[29] Blanchard, E., and Sandu, D., 2007, "A Polynomial Chaos Based Bayesian Approach for Estimating Uncertain Parameters of Mechanical Systems–Part 1: Theoretical Approach," Technical Report No. TR-07-38, Virginia Tech, Blacksburg.

[30] Blanchard, E., and Sandu, D., 2007, "A Polynomial Chaos Based Bayesian Approach for Estimating Uncertain Parameters of Mechanical Systems-Part 2: Applications to Vehicle Systems," Technical Report No. TR-07-39, Virginia Tech, Blacksburg.

[31] Pence, B., Hays, J., Fathy, H., Sandu, C., and Stein, J., 2011, "Vehicle Sprung Mass Estimation for Rough Terrain," International Journal of Vehicle Design, Special Issue on Modeling and Simulation of Ground Vehicle Systems, pp. (To Appear).

[32] Marzouk, Y. M., Najm, H. N., and Rahn, L. A., 2007, "Stochastic Spectral Methods for Efficient Bayesian Solution of Inverse Problems," Journal of Computational Physics, 224 (2), pp. 560-586.

[33] Marzouk, Y., and Xiu, D., 2009, "A Stochastic Collocation Approach to Bayesian Inference in Inverse Problems," Communications in Computational Physics, 6, pp. 826–847.

[34] Smith, A., Monti, A., and Ponci, F., 2007, "Indirect Measurements Via a Polynomial Chaos Observer," IEEE Transactions on Instrumentation and Measurement, 56 (3), pp. 743-752.

[35] Price, D., 2008, "Estimation of Uncertain Vehicle Center of Gravity Using Polynomial Chaos Expansions," Masters thesis, Virginia Tech, Blackburg.

[36] Li, J., and Xiu, D., 2009, "A Generalized Polynomial Chaos Based Ensemble Kalman Filter with High Accuracy," Journal of Computational Physics, 228 (15), pp. 5454-5469.

[37] Saad, G., Ghanem, R., and Masri, S., 2007, "Robust System Identification of Strongly Non-Linear Dynamics Using a Polynomial Chaos Based Sequential Data Assimilation Technique," 6, pp. 6005-13.

[38] Templeton, B., 2009, "A Polynomial Chaos Approach to Control Design," Doctorate thesis, Virginia Tech, Blacksburg.

[39] Smith, A., Monti, A., and Ponci, F., 2006, "Robust Controller Using Polynomial Chaos Theory," Industry Applications Conference, Tampa, FL, 8-12 Oct. 2006, pp. 2511-2517.

[40] Prempraneerach, P., Hover, F., Triantafyllou, M., and Karniadakis, G., 2010, "Uncertainty Quantification in Simulations of Power Systems: Multi-Element Polynomial Chaos Methods," Reliability Engineering & System Safety, 95, pp. 632-646.

[41] Kewlani, G., and Iagnemma, K., 2009, "A Multi-Element Generalized Polynomial Chaos Approach to Analysis of Mobile Robot Dynamics under Uncertainty," pp. 1177-1182.

[42] Park, J., 2007, Industrial Robotics, Programming, Simulation and Applications, Verlag, Croatia, Optimal Motion Planning for Manipulator Arms Using Nonlinear Programming.

[43] Chong Jin, O., and Gilbert, E. G., 1996, "Growth Distances: New Measures for Object Separation and Penetration," Robotics and Automation, IEEE Transactions on, 12 (6), pp. 888-903.

[44] Park, F., Bobrow, J., and Ploen, S., 1995, "A Lie Group Formulation of Robot Dynamics," The International Journal of Robotics Research, 14 (6), pp. 609.

[45] Ploen, S., 1997, "Geometric Algorithms for the Dynamics and Control of Multibody Systems," Doctorate thesis, University Of California, Irvine, CA.

[46] Park, F. C., and Bobrow, J. E., 1994, "A Recursive Algorithm for Robot Dynamics Using Lie Groups," Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on, pp. 1535-1540 vol.2.

[47] Martin, B., and Bobrow, J., 1999, "Minimum-Effort Motions for Open-Chain Manipulators with Task-Dependent End-Effector Constraints," The International Journal of Robotics Research, 18 (2), pp. 213.

[48] Sohl, G. A., and Bobrow, J. E., 2001, "A Recursive Multibody Dynamics and Sensitivity Algorithm for Branched Kinematic Chains," Transactions of the ASME. Journal of Dynamic Systems, Measurement and Control, 123 (Copyright 2002, IEE), pp. 391-9.

[49] Bobrow, J. E., and Sohl, G. A., 2002, "On the Reliable Computation of Optimal Motions for Underactuated Manipulators"

[50] Bobrow, J., Martin, B., Sohl, G., Wang, E., Park, F., and Kim, J., 2001, "Optimal Robot Motions for Physical Criteria," Journal of Robotic systems, 18 (12), pp. 785-795.

[51] Sohl, G., 2000, "Optimal Dynamic Motion Planning for Underactuated Robots," PhD thesis, University of California, Irvine.

[52] Wang, C., Timoszyk, W., and Bobrow, J., 1999, "Weightlifting Motion Planning for a Puma 762 Robot," pp. 480-485.

[53] Wang, C., 2001, "Dynamic Motion Planning for Robot Manipulators Using B-Splines," Doctorate thesis, University Of California, Irvine, CA.

[54] Junggon, K., Jonghyun, B., and Park, F. C., 1999, "Newton-Type Algorithms for Robot Motion Optimization," Intelligent Robots and Systems, 1999. IROS '99. Proceedings. 1999 IEEE/RSJ International Conference on, 3, pp. 1842-1847 vol.3.

[55] Bobrow, J. E., Park, F. C., and Sideris, A., 2006, Fast Motions in Biomechanics and Robotics, Springer Berlin / Heidelberg, Recent Advances on the Algorithmic Optimization of Robot Motion.

[56] Lee, S. H., Kim, J., Park, F. C., Kim, M., and Bobrow, J. E., 2005, "Newton-Type Algorithms for Dynamics-Based Robot Movement Optimization," Robotics, IEEE Transactions on, 21 (4), pp. 657-667.

[57] Xiang, Y., Arora, J., and Abdel-Malek, K., 2009, "Optimization-Based Motion Prediction of Mechanical Systems: Sensitivity Analysis," Structural and Multidisciplinary Optimization, 37 (6), pp. 595-608.

[58] Xiang, Y., Chung, H., Mathai, A., Rahmatalla, S., Kim, J., Marler, T., Beck, S., Yang, J., Arora, J., and Abdel-Malek, K., 2007, "Optimization-Based Dynamic Human Walking Prediction," Optimization, 1, pp. 2489.

[59] Chung, H., Xiang, Y., Mathai, A., Rahmatalla, S., Kim, J., Marler, T., Beck, S., Yang, J., Arora, J., and Abdel-Malek, K., 2007, "A Robust Formulation for Prediction of Human Running," pp. 16–18.

[60] Xiang, Y., Arora, J. S., Rahmatalla, S., and Abdel-Malek, K., 2009, "Optimization-Based Dynamic Human Walking Prediction: One Step Formulation," International Journal for Numerical Methods in Engineering, 79 (6), pp. 667-695.

[61] Xiang, Y., Chung, H.-J., Kim, J., Bhatt, R., Rahmatalla, S., Yang, J., Marler, T., Arora, J., and Abdel-Malek, K., 2010, "Predictive Dynamics: An Optimization-Based Novel Approach for Human Motion Simulation," Structural and Multidisciplinary Optimization, 41 (3), pp. 465-479.

[62] Xiang, Y., Arora, J., and Abdel-Malek, K., 2010, "Physics-Based Modeling and Simulation of Human Walking: A Review of Optimization-Based and Other Approaches," Structural and Multidisciplinary Optimization, 42 (1), pp. 1-23.

[63] Xiang, Y., 2008, "Optimization-Based Dynamic Human Walking Prediction," PhD thesis, University of Iowa, Iowa City, IA.

[64] Kim, H., Wang, Q., Rahmatalla, S., Swan, C., Arora, J., Abdel-Malek, K., and Assouline, J., 2008, "Dynamic Motion Planning of 3d Human Locomotion Using Gradient-Based Optimization," Journal of biomechanical engineering, 130, pp. 031002.

[65] Diehl, M., Ferreau, H., and Haverbeke, N., 2009, "Efficient Numerical Methods for Nonlinear Mpc and Moving Horizon Estimation," Nonlinear Model Predictive Control, pp. 391-417.

[66] Biegler, L. T., 2003, "Optimization of Ode/Dae Constrained Models," Technical Report No.

[67] Park, J., and Park, F., 2006, Advances in Robot Kinematics, Springer Netherlands, A Convex Optimization Algorithm for Stabilizing Whole-Body Motions of Humanoid Robots.

[68] Suleiman, W., Yoshida, E., Laumond, J. P., and Monin, A., 2007, "On Humanoid Motion Optimization," Humanoid Robots, 2007 7th IEEE-RAS International Conference on, pp. 180-187.

[69] Lim, B., Kim, B., Park, F. C., and Hong, D. W., 2009, "Movement Primitives for Three-Legged Locomotion over Uneven Terrain," International Conference on Robotics and Automation, Kobe, Japan

[70] Hays, J., Hong, D., Sandu, C., and Sandu, A., 2010, "Design Optimization for Minimizing Actuation Energy of a Dynamic Tripedal Walking Robot: Revisiting the Double Pendulum," ASME Conference Proceedings, ASME 2010 International Design Engineering Technical Conference, 2010 (44106), pp. 1465-1473.

[71] Lavalle, S., 2006, *Planning Algorithms*, Cambridge Univ Press, New York, NY, USA.

[72] Choset, H., 2005, *Principles of Robot Motion: Theory, Algorithms, and Implementation*, The MIT Press, Cambridge, MA, USA.

[73] Karaman, S., and Frazzoli, E., 2010 (submitted), "Incremental Sampling-Based Algorithms for Optimal Motion Planning," Technical Report No. arXiv:1005.0416v1 [cs.RO], Cornell University, Ithaca, NY.

[74] Barraquand, J., and Ferbach, P., 1995, "Motion Planning with Uncertainty: The Information Space Approach," International Conference on Robotics and Automation, 2, pp. 1341-1348.

[75] Park, W., Liu, Y., Zhou, Y., Moses, M., and Chirikjian, G., 2008, "Kinematic State Estimation and Motion Planning for Stochastic Nonholonomic Systems Using the Exponential Map," Robotica, 26 (04), pp. 419-434.

[76] Erdmann, M., 1984, "On Motion Planning with Uncertainty," Masters thesis, Massachusetts Institute of Technology, Boston.

[77] Kewlani, G., Ishigami, G., and Iagnemma, K., 2009, "Stochastic Mobility-Based Path Planning in Uncertain Environments," pp. 1183-1189.

[78] Hays, J., Sandu, A., Sandu, C., and Hong, D., 2011, "Motion Planning of Uncertain Fully-Actuated Dynamical Systems—an Inverse Dynamics Formulation," ASME IDETC/CIE Conference, Washington, DC, USA

[79] Hays, J., Sandu, A., Sandu, C., and Hong, D., 2011, "Motion Planning of Uncertain Fully-Actuated Dynamical Systems—a Forward Dynamics Formulation," ASME IDETC/CIE Conference, Washington, DC, USA

[80] Hays, J., Sandu, A., Sandu, C., and Hong, D., 2011, "Motion Planning of Uncertain under-Actuated Dynamical Systems—a Hybrid Dynamics Formulation," Proceedings of the ASME 2011 International Mechanical Engieneering Congress & Exposition, Denver, CO, USA

[81] Hays, J., Sandu, A., Sandu, C., and Hong, D., 2011, "Motion Planning of Uncertain Ordinary Differential Equation Systems," Technical Report No. TR-11-04, Virginia Tech, Blacksburg, VA, USA.

[82] Stewart, D., and Anitescu, M., 2010, "Optimal Control of Systems with Discontinuous Differential Equations," Numerische Mathematik, 114 (4), pp. 653-695.

[83] Zefran, M., and Kumar, V., 1995, "Optimal Control of Systems with Unilateral Constraints," pp. 2695-2695.

[84] Tassa, Y., and Todorov, E., 2010, "Stochastic Complementarity for Local Control of Discontinuous Dynamics"

[85] Fathy, H., Papalambros, P., and Ulsoy, A., 2004, "On Combined Plant and Control Optimization," American control Conference, Arlington, VA, 2001, pp. 1864-1869.

[86] Alyaqout, S., Papalambros, P., and Ulsoy, A., 2007, "Combined Design and Robust Control of a Vehicle Passive/Active Suspension," European Control Conference, Kos, Greece

[87] Alyaqout, S., Papalambros, P., and Ulsoy, A., 2007, "Coupling in Design and Robust Control Optimization," European Control Conference, Kos, Greece

[88] Ravichandran, T., Wang, D., and Heppler, G., "Simultaneous Plant-Controller Design Optimization of a Two-Link Planar Manipulator," Mechatronics, 16 (3-4), pp. 233-242.

[89] Gobbi, M., and Mastinu, G., 2001, "Analytical Description and Optimization of the Dynamic Behaviour of Passively Suspended Road Vehicles," Journal of Sound and Vibration, 245 (3), pp. 457-481.

[90] Verros, G., Natsiavas, S., and Papadimitriou, C., 2005, "Design Optimization of Quarter-Car Models with Passive and Semi-Active Suspensions under Random Road Excitation," Journal of Vibration and Control, 11 (5), pp. 581-606.

[91] He, Y., and Mcphee, J., 2005, "Multidisciplinary Design Optimization of Mechatronic Vehicles with Active Suspensions," Journal of Sound and Vibration, 283 (1-2), pp. 217-241.

[92] He, Y., and Mcphee, J., 2007, "Application of Optimisation Algorithms and Multibody Dynamics to Ground Vehicle Suspension Design," International Journal of Heavy Vehicle Systems, 14 (2), pp. 158-192.

[93] Alkhatib, R., Nakhaie Jazar, G., and Golnaraghi, M., 2004, "Optimal Design of Passive Linear Suspension Using Genetic Algorithm," Journal of Sound and Vibration, 275 (3-5), pp. 665-691.

[94] Jazar, R. N., 2008, Vehicle Dynamics: Theory and Application, Springer US, Suspension Optimization.

[95] Hrovat, D., 1993, "Applications of Optimal Control to Advanced Automotive Suspension Design," Journal of Dynamic Systems, Measurement, and Control, 115 (2B), pp. 328-342.

[96] Georgiou, G., Verros, G., and Natsiavas, S., 2007, "Multi-Objective Optimization of Quarter-Car Models with a Passive or Semi-Active Suspension System," Vehicle System Dynamics: International Journal of Vehicle Mechanics and Mobility, 45 (1), pp. 77 - 92.

[97] Segla, S., and Reich, S., 2007, "Optimization and Comparison of Passive, Active, and Semi-Active Vehicle Suspension Systems," 12th IFToMM World Congress, Besancon, France

[98] Baumal, A. E., Mcphee, J. J., and Calamai, P. H., 1998, "Application of Genetic Algorithms to the Design Optimization of an Active Vehicle Suspension System," Computer methods in applied mechanics and engineering, 163 (1-4), pp. 87-94.

[99] He, Y., and Mcphee, J., 2005, "Multidisciplinary Optimization of Multibody Systems with Application to the Design of Rail Vehicles," Multibody System Dynamics, 14 (2), pp. 111-135.

[100] He, Y., and Mcphee, J., 2005, "A Design Methodology for Mechatronic Vehicles: Application of Multidisciplinary Optimization, Multibody Dynamics and Genetic Algorithms," Vehicle System Dynamics: International Journal of Vehicle Mechanics and Mobility, 43 (10), pp. 697 - 733.

[101] Good, and Mcphee, 1999, "Dynamics of Mountain Bicycles with Rear Suspensions: Modelling and Simulation," Sports Engineering (International Sports Engineering Association), 2 (3), pp. 129.

[102] Good, C., and Mcphee, J., 2000, "Dynamics of Mountain Bicycles with Rear Suspensions: Design Optimization," Sports Engineering (International Sports Engineering Association), 3 (1), pp. 49-55.

[103] Hays, J., Sandu, A., Sandu, C., and Hong, D., 2011, "Parametric Design Optimization of Uncertain Ordinary Differential Equation Systems," Proceedings of the ASME 2011 International Mechanical Engieneering Congress & Exposition, Denver, CO, USA

[104] Hays, J., Sandu, A., Sandu, C., and Hong, D., 2011, "Parametric Design Optimization of Uncertain Ordinary Differential Equation Systems," Technical Report No. TR-11-06, Virginia Tech, Blacksburg, VA, USA.

[105] Chase, K. W., 1999, Dimensioning and Tolerancing Handbook, McGraw-Hill Professional, Minimum-Cost Tolerance Allocation.

[106] Chase, K. W., 1999, "Tolerance Allocation Methods for Designers," ADCATS Report, 99 (6), pp. 1-28.

[107] Chase, K. W., Greenwood, W. H., Loosli, B. G., and Hauglund, L. F., 1990, "Least Cost Tolerance Allocation for Mechanical Assemblies with Automated Process Selection," Manufacturing Review, 3 (1), pp. 49-59.

[108] Barraja, M., and Vallance, R. R., 2002, "Tolerance Allocation for Kinematic Couplings"

[109] Merkley, K. G., 1998, "Tolerance Analysis of Compliant Assemblies," Doctorate thesis, Brigham Young University, Provo, UT.

[110] Krishnaswami, P., and Kelkar, A. G., 2003, "Optimal Design of Controlled Multibody Dynamic Systems for Performance, Robustness and Tolerancing," Engineering with Computers, 19 (1), pp. 26-34.

[111] Arenbeck, H., Missoum, S., Basudhar, A., and Nikravesh, P., 2010, "Reliability-Based Optimal Design and Tolerancing for Multibody Systems Using Explicit Design Space Decomposition," Journal of Mechanical Design, 132 (2), pp. 021010-11.

[112] Lee, S. J., Gilmore, B. J., and Ogot, M. M., 1993, "Dimensional Tolerance Allocation of Stochastic Dynamic Mechanical Systems through Performance and Sensitivity Analysis," Journal of Mechanical Design, 115 (3), pp. 392-402.

[113] Rout, B., and Mittal, R., 2010, "Simultaneous Selection of Optimal Parameters and Tolerance of Manipulator Using Evolutionary Optimization Technique," Structural and Multidisciplinary Optimization, 40 (1), pp. 513-528.

[114] Hays, J., Sandu, A., Sandu, C., and Hong, D., 2011, "Simultaneous Optimal Uncertainty Apportionment and Robust Design Optimization of Systems Governed by Ordinary Differential Equations," Technical Report No. TR-11-15, Virginia Tech, Blacksburg, VA, USA.

[115] Greenwood, D., 2003, *Advanced Dynamics*, Cambridge University Press, Cambridge, UK.

[116] Greenwood, D., 1997, *Classical Dynamics*, Dover Publications, Mineola, NY.

[117] Meirovitch, L., 2004, *Methods of Analytical Dynamics*, Dover Publications, Mineola, NY.

[118] Greenwood, D., 1988, *Principles of Dynamics*, Prentice-Hall Englewood Cliffs, NJ.

[119] José, J., and Saletan, E., 1998, *Classical Dynamics: A Contemporary Approach*, Cambridge University Press, Cambridge, UK.

[120] Haug, E. J., 1989, *Computer Aided Kinematics and Dynamics of Mechanical Systems. Vol. 1: Basic Methods*, Allyn & Bacon, Inc., Boston, Massachusetts.

[121] Featherstone, R., 2008, *Rigid Body Dynamics Algorithms*, Springer-Verlag, New York.

[122] Son, W., Kim, K., Amato, N., and Trinkle, J., 2004, "A Generalized Framework for Interactive Dynamic Simulation for Multirigid Bodies," IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 34 (2), pp. 912-924.

[123] Bhalerao, K., Anderson, K., and Trinkle, J., 2009, "A Recursive Hybrid Time-Stepping Scheme for Intermittent Contact in Multi-Rigid-Body Dynamics," Journal of Computational and Nonlinear Dynamics, 4, pp. 041010.

[124] Trinkle, J., 2003, "Formulation of Multibody Dynamics as Complementarity Problems," ASME 2003 Design Engineering Technical Conferences and Computers and Information in Ingineering Conference, Chicago, Illinois, USA, pp. 361–370.

[125] Stewart, D. E., and Trinkle, J. C., 1996, "An Implicit Time-Stepping Scheme for Rigid Body Dynamics with Inelastic Collisions and Coulomb Friction," International Journal for Numerical Methods in Engineering, 39 (15), pp. 2673-2691.

[126] Chakraborty, N., Berard, S., Akella, S., and Trinkle, J., 2007, "An Implicit Time-Stepping Method for Multibody Systems with Intermittent Contact," Technical Report No. Rensselaer Polytechnic Institute, Troy, NY.

[127] Chakraborty, N., Berard, S., Akella, S., and Trinkle, J., 2007, "An Implicit Time-Stepping Method for Quasi-Rigid Multibody Systems with Intermittent Contact," ASME 2007 International Design Engineering Technical Conference & Computers and Information in Engineering Conference, Las Vegas, Nevada, USA

[128] Stewart, D., and Trinkle, J., 2000, "An Implicit Time-Stepping Scheme for Rigid Body Dynamics with Coulomb Friction," 1, pp. 162-169.

[129] Anitescu, M., 2003, "A Fixed Time-Step Approach for Multibody Dynamics with Contact and Friction," Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on, 4, pp. 3725-3731 vol.3.

[130] Tasora, A., and Anitescu, M., 2008, Multibody Dynamics, Springer Netherlands, A Fast Ncp Solver for Large Rigid-Body Problems with Contacts, Friction, and Joints.

[131] Hart, G. D., and Anitescu, M., 2003, "A Hard-Constraint Time-Stepping Approach for Rigid Multibody Dynamics with Joints, Contact, and Friction," Proceedings of the 2003 conference on Diversity in computing, Atlanta, Georgia, USA

[132] Anitescu, M., and Potra, F. A., 2002, "A Time-Stepping Method for Stiff Multibody Dynamics with Contact and Friction," International Journal for Numerical Methods in Engineering, 55 (7), pp. 753-784.

[133] Anitescu, M., and Hart, G. D., 2004, "A Constraint-Stabilized Time-Stepping Approach for Rigid Multibody Dynamics with Joints, Contact and Friction," International Journal for Numerical Methods in Engineering, 60 (14), pp. 2335-2371.

[134] Stewart, D., 1990, "A High Accuracy Method for Solving Odes with Discontinuous Right-Hand Side," Numerische Mathematik, 58 (1), pp. 299-328.

[135] Yamane, K., and Nakamura, Y., 2008, "A Numerically Robust Lcp Solver for Simulating Articulated Rigid Bodies in Contact," Robotics: Science and Systems, Zurich, Switzerland

[136] Ellekilde, L.-P., and Petersen, H. G., 2006, "Simulating Non-Holonomic Constraints within the Lcp-Based Simulation Framework," International Journal for Numerical Methods in Engineering, 67 (10), pp. 1445-1466.

[137] Dingliana, J., and O'sullivan, C., 2005, "A Voxel-Based Approach to Approximate Collision Handling," Journal of Graphics, GPU, & Game Tools, 10 (4), pp. 33-48.

[138] Xiushan, Z., Chanle, W., Ke'rong, B., and Shui, C., "Collision Graph-Based Multi-Body Simultaneous Collision Detection Mechanism," 670, pp. 160.

[139] Baraff, D., 1994, "Fast Contact Force Computation for Nonpenetrating Rigid Bodies," Proceedings of the 21st annual conference on Computer graphics and interactive techniques, pp. 23-34.

[140] Drumwright, E., 2008, "A Fast and Stable Penalty Method for Rigid Body Simulation," IEEE Transactions on Visualization and Computer Graphics, 14 (1), pp. 231-240.

[141] Mirtich, B., and Canny, J., 1995, "Impulse-Based Simulation of Rigid Bodies," Proceedings of the 1995 symposium on Interactive 3D graphics, Monterey, California, United States

[142] Mirtich, B., 1996, "Impulse-Based Dynamic Simulation of Rigid Body Systems," Doctorate thesis, University of California, Berkley, CA.

[143] Bender, J., and Schmitt, A., 2006, "Fast Dynamic Simulation of Multi-Body Systems Using Impulses," Proc. VRIPhys, pp. 81-90.

[144] Bender, J., 2007, "Impulse-Based Dynamic Simulation in Linear Time," Comput. Animat. Virtual Worlds, 18 (4-5), pp. 225-233.

[145] Weinstein, R., Teran, J., and Fedkiw, R., 2006, "Dynamic Simulation of Articulated Rigid Bodies with Contact and Collision," Visualization and Computer Graphics, IEEE Transactions on, 12 (3), pp. 365-374.

[146] Schmitke, C., and Mcphee, J., 2008, "Using Linear Graph Theory and the Principle of Orthogonality to Model Multibody, Multi-Domain Systems," Adv. Eng. Inform., 22 (2), pp. 147-160.

[147] Murray, R., Li, Z., Sastry, S., and Sastry, S., 1994, *A Mathematical Introduction to Robotic Manipulation*, CRC Press, Inc, Boca Raton, FL, USA.

[148] Baraff, D., 1996, "Linear-Time Dynamics Using Lagrange Multipliers," Computer Graphics, New Orleans, LA, pp. 137-146.

[149] Mukherjee, R., and Anderson, K., 2007, "Orthogonal Complement Based Divide-and-Conquer Algorithm for Constrained Multibody Systems," Nonlinear Dynamics, 48 (1), pp. 199-215.

[150] Yamane, K., and Nakamura, Y., 2009, "Comparative Study on Serial and Parallel Forward Dynamics Algorithms for Kinematic Chains*," Int. J. Rob. Res., 28 (5), pp. 622-629.

[151] Featherstone, R., 2001, "The Acceleration Vector of a Rigid Body," The International Journal of Robotics Research, 20 (11), pp. 841-846.

[152] Featherstone, R., and Orin, D., 2000, "Robot Dynamics: Equations and Algorithms," 1, pp. 826-834.

[153] Featherstone, R., 1999, "A Divide-and-Conquer Articulated-Body Algorithm for Parallel O(Log(N)) Calculation of Rigid-Body Dynamics. Part 2: Trees, Loops, and Accuracy," The International Journal of Robotics Research, 18 (9), pp. 876-892.

[154] Featherstone, R., 1999, "A Divide-and-Conquer Articulated-Body Algorithm for Parallel O(Log(N)) Calculation of Rigid-Body Dynamics. Part 1: Basic Algorithm," The International Journal of Robotics Research, 18 (9), pp. 867-875.

[155] Law, M., 1996, "Multivariate Statistical Analysis of Assembly Tolerance Specifications," MS thesis, Brigham Young University, Provo, UT, USA.

[156] Nikravesh, P. E., 2004, Product Engineering, Springer Verilag, Berlin, Germany, An Overview of Several Formulations for Multibody Dynamics.

[157] Brenan, K. E., Campbell, S. L., and Petzold, L. R., 1996, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*, Society for Industrial and Applied Mathematics, Philadelphia, PA.

[158] Leyffer, S., and Mahajan, A., 2010, "Nonlinear Constrained Optimization: Methods and Software," Technical Report No. ANL/MCS-P1729-0310, Argonne National Laboratory, Argonne, IL.

[159] Boyd, S., and Vandenberghe, L., 2004, *Convex Optimization*, Cambridge University Press, Cambridge, UK.

[160] Bazaraa, M., Sherali, H., and Shetty, C., 2006, *Nonlinear Programming: Theory and Algorithms*, John Wiley and Sons, Hoboken, NJ.

[161] Rao, S., 2009, *Engineering Optimization: Theory and Practice*, John Wiley and Sons, Hoboken, NJ.

[162] Nelder, J., and Mead, R., 1965, "A Simplex Method for Function Minimization," The computer journal, 7 (4), pp. 308.

[163] Mathematica, 2009, Wolfram Research.

[164] Jones, D. R., Perttunen, C. D., and Stuckman, B. E., 1993, "Lipschitzian Optimization without the Lipschitz Constant," Journal of Optimization Theory and Applications, 79 (1), pp. 157-181.

[165] Gablonsky, J. M., and Kelley, C. T., 2001, "A Locally-Biased Form of the Direct Algorithm," Journal of Global Optimization, 21 (1), pp. 27-37.

[166] Price, K., Storn, R., and Lampinen, J., 2005, *Differential Evolution: A Practical Approach to Global Optimization*, Springer Verlag, Berlin, Germany.

[167] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P., 1983, "Optimization by Simulated Annealing," Science, 220 (4598), pp. 671-680.

[168] Neumaier, A., Shcherbina, O., Huyer, W., and Vinkó, T., 2005, "A Comparison of Complete Global Optimization Solvers," Mathematical Programming, 103 (2), pp. 335-356.

[169] Neumaier, A., 2005, Introduction to Global Optimization, http://www.mat.univie.ac.at/~neum/glopt/intro.html

[170] Horst, R., Pardalos, P., and Thoai, N., 2000, *Introduction to Global Optimization*, Springer Verlag, Berlin, Germany.

[171] Khompatraporn, C., Pintér, J. D., and Zabinsky, Z. B., 2005, "Comparative Assessment of Algorithms and Software for Global Optimization," Journal of Global Optimization, 31 (4), pp. 613-633.

[172] Antoniou, A., and Lu, W., 2007, *Practical Optimization: Algorithms and Engineering Applications*, Springer-Verlag, New York.

[173] Biegler, L. T., and Grossmann, I. E., 2004, "Retrospective on Optimization," Computers & Chemical Engineering, 28 (8), pp. 1169-1192.

[174] Miettinen, K., 1999, *Nonlinear Multiobjective Optimization*, Springer, Berlin, Germany.

[175] Cheng, H., and Sandu, A., 2007, "Numerical Study of Uncertainty Quantification Techniques for Implicit Stiff Systems," Proceedings of the 45-th ACM Southeast Conference, Winston-Salem, NC, USA, pp. 367-372.

[176] Cheng, H., and Sandu, A., 2009, "Uncertainty Quantification in 3d Air Quality Models Using Polynomial Chaoses," Environmental Modeling and Software, 24 (8), pp. 917-925.

[177] Cheng, H., and Sandu, A., 2009, "Uncertainty Apportionment for Air Quality Forecast Models," Proceedings of 24th Annual ACM Symposium on Applied Computing (SAC-2009), Computational Sciences Track, Honolulu, HI, USA, pp. 956-960.

[178] Cheng, H., and Sandu, A., 2010, "Collocation Least-Squares Polynomial Chaos Method," Proceedings of the 2010 Spring Simulation Multiconference (SpringSim'10), High Performance Computing Symposium (HPC-2010), Orlando, FL, USA, pp. 80.

[179] Shene, C. K., 1997, Introduction to Computing with Geometry Notes, http://www.cs.mtu.edu/~shene/COURSES/cs3621/NOTES/

[180] Piegl, L. A., and Tiller, W., 1997, *The Nurbs Book*, Springer Verlag, Berlin, Germany.

[181] Adcats, 2006, One-Way Clutch Verification, http://adcats.et.byu.edu/WWW/ADCATS/Example_Problems/ProE_Verify/2clutch/2clutch.html