SIAM J. Sci. Comput.
Vol. 41, No. 4, pp. A2065–A2087

NUMERICAL TRAJECTORY OPTIMIZATION FOR STOCHASTIC MECHANICAL SYSTEMS[*]

GEORGE I. BOUTSELIS[†], YUNPENG PAN[†‡], AND EVANGELOS A. THEODOROU[†]

Abstract. In this paper we develop a novel optimal control framework for uncertain mechanical systems. Our work extends differential dynamic programming and handles uncertainty through generalized polynomial chaos (gPC) theory. The obtained scheme is able to influence the probabilistic evolution of nonlinear systems with stochastic model parameters. Its scalable, fast-converging nature plays a key role when dealing with gPC expansions of high-dimensional problems. Based on Lagrangian principles, we also prove that variational integrators can be designed to properly propagate and linearize gPC representations in discrete time. This observation allows us to further improve the efficiency of our trajectory-optimization methodology. Numerical simulations support the aforementioned arguments, while demonstrating the benefits of our approach over standard optimization methods. Last but not least, a complete analysis is provided that studies the properties of the proposed algorithm.

Key words. trajectory optimization, polynomial chaos, differential dynamic programming, discrete mechanics

AMS subject classifications. 41A10, 49L20, 70H03, 93E20

DOI. 10.1137/17M116272X

1. Introduction.

Applying efficient control schemes on autonomous systems remains a key challenge for the scientific community. Towards this goal, optimal control theory has been widely employed, with applications both in simulated and real environments (see, e.g., [1, 7, 26, 30]). Mathematically, this involves optimizing a performance criterion relevant to the task under consideration, while respecting constraints of the involved systems. Unfortunately, standard optimal control approaches become inefficient when modeling accuracy degrades. To this end, several stochastic optimal control algorithms have been proposed that rely on various ways of uncertainty representation.

One class of optimal control methods represents uncertainty by treating the dynamics as stochastic differential equations. Some popular approaches include iterative linearization methods such as iterative linear quadratic Gaussian control [31] and stochastic differential dynamic programming, as well as sampling methods like path integral control [18]. This way of modeling, however, cannot capture stochasticity directly in the internal parameters, from which many times uncertainty stems. In addition, continuous-time white noise signals possess infinite energy and, therefore, do not exist in the physical world.

Recently, probabilistic trajectory-optimization algorithms [7, 26] have been developed based on Gaussian process regression [28]. Gaussian processes (GPs) provide a way to represent unknown functions by storing data collected from a system and

[*]Submitted to the journal's Methods and Algorithms for Scientific Computing section December 26, 2017; accepted for publication (in revised form) February 21, 2019; published electronically July 2, 2019. Preliminary results of this work have been published in the American Control Conference 2016 and the Conference on Decision and Control 2016 [3, 5, 4].
https://doi.org/10.1137/17M116272X
Funding: This work was supported by NSF-NRI grant 1426945. The first author's work was supported by the A. S. Onassis Foundation.

[†]Department of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0150 (gbouts@gatech.edu, ypan37@gatech.edu, evangelos.theodorou@gatech.edu).
[‡]Magna Autonomous Systems LLC, Palo Alto, CA 94304.

performing inference on them. Due to their probabilistic nature, GP-based methods can be employed for learning control on unknown systems, while using a relatively low number of interactions during training. Nevertheless, most relevant approaches achieve tractability in the inference phase by treating the state as a Gaussian random variable, and then relying on moment matching methods (see, e.g., [7]). Since the Gaussianity assumption is rarely satisfied for nonlinear systems, this may lead to erroneous representations.

One method widely used for quantifying uncertainty is polynomial chaos theory. Wiener introduced polynomial chaos [34] and used it to decompose stochastic processes into a convergent series of Hermite polynomials. Xiu and Karniadakis in [36] extended this concept and introduced the generalized polynomial chaos (gPC) framework. In their work, various continuous and discrete distributions were modeled using orthogonal polynomials from the Askey scheme, and L^2 convergence was guaranteed. Polynomial chaos theory has been successfully applied in various fields such as solid mechanics [12] and fluid dynamics [14]. However, works on control-related problems are rather limited. Dutta and Bhattacharya in [9] designed nonlinear estimators which proved to be beneficial when measurements are infrequent. Hover and Triantafyllou utilized gPC as a tool to analyze the stability of a stochastic bilinear system [15]. Moreover, Fisher and Bhattacharya in [11] proposed a stochastic version of the linear-quadratic regulator.

In this work we consider dynamic systems influenced by uncertain internal parameters and initial states. The exact values of these quantities will be unknown, but we will assume we have access to their underlying statistics. The gPC framework will be employed to transform the, originally, stochastic system into a purely deterministic one. To solve the obtained problems, we propose an extension of differential dynamic programming (DDP) [16, 29] and call the developed algorithm gPC-DDP. Our iterative scheme is capable of controlling the probabilistic behavior of the trajectory in an optimization setting.

The performance of the proposed methodology is further improved by incorporating the concept of variational integrators (VIs). VIs are a class of numerical time-stepping methods derived from discretization of Hamilton's principle [19, 24]. They have been shown to outperform standard numerical integration schemes (e.g., Euler or Runge–Kutta methods) due to their long-term energy preserving properties. One practical consequence is their insensitivity to both time-step and horizon selection [24, 32]. In addition, their available structured linearization renders them ideal candidates to use when numerically solving optimal control problems [17]. Inspired by the work in [27], we will develop variational integration schemes for gPC-based dynamics and incorporate them into our control methodology.

1.1. Contributions.

Our contributions can be summarized as follows:

- A numerical methodology for controlling polynomial chaos representations of stochastic systems is derived, which relies on DDP. Its inherent scalability allows one to obtain tractable solutions, even when dealing with high-dimensional gPC representations.
- We prove that gPC expansions of mechanical systems obey Lagrangian mechanics principles. This is accomplished by providing explicit expressions for the associated Lagrangian function and nonconservative forces (e.g., control inputs, or dissipation). Consequently, VIs can be designed to obtain faithful discrete representations of stochastic systems. This further implies that one can reduce the computational complexity of our scheme by allowing coarse

discretizations of the time horizon.

- We provide a complete convergence proof for our optimal control algorithm by considering problems with generic running and terminal cost terms. Our analysis, therefore, generalizes prior theoretical work on the properties of DDP-based algorithms.
- We use simulated examples to validate the applicability of the developed control scheme and to compare its performance with off-the-shelf optimization methods.

The remainder of this paper is organized as follows: Section 2 states the problem we will be tackling. Section 3 includes preliminaries on gPC theory and discrete mechanics. In section 4, we discuss polynomial chaos representations of dynamic systems, for which we also develop a VI. Section 5 studies polynomial chaos formulations of stochastic optimal control problems and derives the gPC-DDP framework. Theoretical guarantees of this scheme are provided in subsection 5.4. Section 6 includes a set of simulated examples that give further insight into the behavior of our algorithm. Lastly, section 7 discusses possible extensions of this work, and section 8 is the conclusion.

2. Problem formulation.

We consider optimal control problems of the form

$$\begin{aligned} &\min_u \int_{t_0}^{t_f} \mathcal{L}(\mathcal{M}(t), u, t)\mathrm{d}t + \mathcal{F}(\mathcal{M}(t_f), t_f) \\ &\text{s.t.}\quad \dot{x}(t) = f(x(t), u(t), t; \lambda^p), \quad x(t_0) = \bar{x}^0(\lambda^0). \end{aligned} \tag{1}$$

Here, $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^m$ is the control input, and $\mathcal{L}$, $\mathcal{F}$ denote the running and terminal costs. In addition, $\lambda^p \in \mathbb{R}^{d_p}$, $\lambda^0 \in \mathbb{R}^{d_0}$ are independent random variables associated with uncertainty in model parameters and initial states, respectively. We assume that the statistics of these latter variables are known a priori. Finally, $\mathcal{M} := (\mathcal{M}_1, \dots, \mathcal{M}_n)^\top$ contains the statistical moments of our state, with $(\mathcal{M}_i)_j$ being the jth (central) moment of x_i, and $\mathcal{M}_i := ((\mathcal{M}_i)_1, \dots, (\mathcal{M}_i)_j, \dots, (\mathcal{M}_i)_N)$.

This paper will focus on discrete counterparts of (1). We will find in section 5 that such stochastic optimal control problems (usually referred to as moment-based control problems; see, e.g., [37, 38]) can be recast into deterministic ones through gPC theory. Lastly, we emphasize that searching for global solutions of (1) becomes computationally infeasible for the general case. Hence, our goal is to develop tractable numerical methods, possibly at the expense of global optimality.

3. Preliminaries.

3.1. Wiener–Askey polynomial chaos.

In this subsection we review the basics of gPC theory. More details can be found in [35].

Let $(\Omega, \mathcal{G}, P)$ be a probability space such that Ω is the sample space, $\mathcal{G}$ is the σ-field of Ω, and P is a corresponding probability measure. Let also $\xi(\omega) \in \mathbb{R}^d$ be a continuous random variable with mutually independent components for each $\omega \in \Omega$. We define the weighted L^2_ρ space as $L^2_\rho := \{\mathsf{f} : \mathcal{D} \to \mathbb{R} | \int_\mathcal{D} \mathsf{f}^2(\xi)\rho(\xi)\mathrm{d}\xi < \infty\}$, where $\mathcal{D}$ is the support of ξ, and $\rho(\xi)$ denotes the density function associated with P. Thus, L^2_ρ denotes the space of mean-square integrable functions.

The *polynomial chaos expansion* of a function $\mathsf{f} \in L^2_\rho$ is given by the series $\mathsf{f}(\xi) = \sum_{j=0}^\infty \mathsf{f}_j \phi_j(\xi)$. Specifically, $\{\mathsf{f}_j,\ j \in \mathbb{Z}_{\geq 0}\}$ denote the polynomial chaos coefficients, while the polynomials $\{\phi_j(\xi),\ j \in \mathbb{Z}_{\geq 0}\}$ satisfy $\int_\mathcal{D} \phi_j(\xi)\phi_i(\xi)\rho(\xi)\mathrm{d}\xi = 0$ for each $i \neq j$. In practice one has to truncate the expansion, as shown in (2). Therein,

Table 1

Correspondence between distributions and orthogonal polynomials from the Askey scheme (α, β denote parameters of the corresponding density functions).

Distribution	Weight function	Polynomials	Domain
Gaussian	$e^{-\xi^2/2}$	Hermite	$(-\infty,\infty)$
Uniform	1	Legendre	$[-1,1]$
Gamma	$\xi^\alpha e^{-\xi}$	Laguerre	$[0,\infty)$
Beta	$(1-\xi)^\alpha(1+\xi)^\beta$	Jacobi	$[-1,1]$

we also give an expression for computing the jth polynomial chaos coefficient. This method relies on the orthogonality of $\{\phi_j\}$ and is called the Galerkin projection:

$$\mathsf{f}(\xi) \approx \textstyle\sum_{j=0}^{K} \mathsf{f}_j \phi_j(\xi), \quad \mathsf{f}_j = \frac{\int_{\mathcal{D}} \mathsf{f}(\xi)\phi_j(\xi)\rho(\xi)\mathrm{d}\xi}{\int_{\mathcal{D}} \phi_j^2(\xi)\rho(\xi)\mathrm{d}\xi}. \tag{2}$$

The series above can be viewed as an orthogonal projection of f onto the space spanned by $\{\phi_j,\ j=0,\dots,K\}$. Each ϕ_j is written as a product of univariate orthogonal polynomials. When these ϕ_j's are chosen to have total degree less than or equal to r, the number of coefficients will be given by $K = \frac{(r+d)!}{r!d!} - 1$. As shown in [35], this constitutes the best polynomial approximation in the L^2_ρ norm.

Xiu and Karniadakis developed the generalized polynomial chaos (gPC) framework in [36]. More precisely, they showed that when $\rho(\xi)$ is of a certain type, one can naturally select an appropriate orthogonal set, $\{\phi_j\}$, from the Askey scheme. Table 1 shows this correspondence in detail. Each set $\{\phi_j\}$ forms a complete orthogonal basis in the Hilbert space determined by $\rho(\xi)$.

Lastly, in the remainder of this paper we will often write

$$\langle \mathsf{f} \rangle := \mathbb{E}[\mathsf{f}] = \textstyle\int_{\mathcal{D}} \mathsf{f}(\xi)\rho(\xi)\mathrm{d}\xi, \quad \langle \mathsf{f}, \mathsf{g} \rangle := \mathbb{E}[\mathsf{fg}] = \int_{\mathcal{D}} \mathsf{f}(\xi)\mathsf{g}(\xi)\rho(\xi)\mathrm{d}\xi, \quad \dots .$$

3.2. Discrete mechanics and variational integrators.

Here, we present fundamental concepts from Lagrangian mechanics in the continuous- and discrete-time domains. We will consider the case where nonconservative forces are applied, since it is more relevant to practical applications.

The continuous Lagrange–d'Alembert principle: Given a finite-dimensional system, let $q := (q_1,\dots,q_N)^\top \in \mathbb{R}^N$, $F := (F_1,\dots,F_N)^\top \in \mathbb{R}^N$, and $L(q,\dot q) \in \mathbb{R}$ denote its generalized position coordinates, nonconservative forces, and Lagrangian function, respectively. The Lagrange–d'Alembert principle states that, given a time horizon $[t_0, t_f]$, admissible trajectories must satisfy $\delta \int_{t_0}^{t_f} L(q,\dot q)\mathrm{d}t + \int_{t_0}^{t_f} F(q,\dot q,u)\delta q \mathrm{d}t = 0$. From this, one can obtain the *Euler–Lagrange equations* of the system [24]

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot q_i} - \frac{\partial L}{\partial q_i} = F_i, \quad i = 1,\dots,N. \tag{3}$$

The continuous Pontryagin–d'Alembert principle: To proceed, let $v \in \mathbb{R}^N$, $p \in \mathbb{R}^N$ denote the generalized velocity and momentum coordinates, respectively. The Pontryagin–d'Alembert principle connects the Lagrangian and Hamiltonian points of view and enforces the condition $\delta \int_{t_0}^{t_f} \big(L(q,v) + p\cdot(\dot q - v)\big)\mathrm{d}t + \int_{t_0}^{t_f} F(q,\dot q,u)\delta q\mathrm{d}t = 0$. Manipulation of this law yields [19]

$$v_i = \dot q_i, \quad p_i = \frac{\partial L}{\partial v_i}, \quad \dot p_i = \frac{\partial L}{\partial q_i} + F_i, \quad i = 1,\dots,N. \tag{4}$$

Equation (4) is equivalent to (3) and will be used in section 4 to build the Lagrangian function and nonconservative forces for gPC representations of mechanical systems.

Discrete Lagrangian mechanics: We now consider discrete versions of (3). Let q^k, u^k denote the discrete pose and control input at instant t_k, and let Δt be the (fixed) time step. We define the *discrete Lagrangian*, $L_d : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$, such that $L_d(q^k, q^{k+1}) \simeq \int_{t_k}^{t_{k+1}} L(q(s), \dot{q}(s))\mathrm{d}s$. This quantity can be numerically determined via any quadrature method [13]. From a computational standpoint, one-point rules combine accuracy and efficiency. Specifically, we write

$$L_d(q^k, q^{k+1}) = L\Big((1-\zeta)q^k + \zeta q^{k+1}, \frac{q^{k+1}-q^k}{\Delta t}\Big)\Delta t, \tag{5}$$

such that $\zeta \in [0,1]$. When $\zeta \equiv 1/2$, the quadrature corresponds to the midpoint rule and is second-order accurate. A more involved expression can be used to increase the order of the resulting integrator [24].

In a similar manner, the continuous nonconservative forces are approximated by their left and right discrete counterparts as follows: $\int_{t_k}^{t_{k+1}} F(q(s), \dot{q}(s), u(s))\delta q \mathrm{d}s \simeq F_d^-(q^k, q^{k+1}, u^k)\delta q^k + F_d^+(q^k, q^{k+1}, u^k)\delta q^{k+1}$. One can typically choose [17]

$$F_d^-(q^k, q^{k+1}, u^k) = F\left(\frac{q^k + q^{k+1}}{2}, \frac{q^{k+1}-q^k}{\Delta t}, u^k\right)\Delta t, \quad F_d^+(q^k, q^{k+1}, u^k) = 0. \tag{6}$$

Plugging the above approximations into the Lagrange–d'Alembert principle gives the discrete Euler–Lagrange (DEL) equations [24]:

$$0 = p^k + D_1 L_d(q^k, q^{k+1}) + F_d^-(q^k, q^{k+1}, u^k), \tag{7a}$$

$$p^{k+1} = D_2 L_d(q^k, q^{k+1}) + F_d^+(q^k, q^{k+1}, u^k). \tag{7b}$$

In this way, we obtain a VI for propagating the discrete dynamics forward in time. Specifically, given (q^k, p^k), (7a) is solved implicitly for q^{k+1}, while (7b) determines p^{k+1} explicitly. Here, $D_i L_d(\cdot)$ denotes the partial derivative of L_d with respect to its ith argument, and p^k can be viewed as the discrete momentum at $t \equiv t_k$. It has been shown that variational integration methods generally outperform schemes that discretize the equations of motion directly [19, 24].

4. Polynomial chaos representations of dynamical systems.

This section uses polynomial chaos theory to express stochastic dynamics in a deterministic fashion. Its main contribution lies in computing the Lagrangian function and nonconservative forces for gPC expansions of generalized coordinates. This allows one to develop variational integration methods and efficiently propagate stochastic mechanical systems in discrete time.

4.1. Dynamics of polynomial chaos coefficients.

Consider the stochastic dynamics of problem (1), restated below for convenience:

$$\dot{x}(t) = f(x(t), u(t), t; \lambda^p), \quad x(t_0) = \bar{x}^0(\lambda^0), \tag{8}$$

with $\lambda^p \in \mathbb{R}^{d_p}$, $\lambda^0 \in \mathbb{R}^{d_0}$ representing uncertainty in the internal dynamics parameters and initial states, respectively, and $x \in \mathbb{R}^n$ being the state of the system. In what follows, we consider the controls $u \in \mathbb{R}^m$ to be deterministic, and each random quantity to be mean-square integrable.

First, we will expand λ^p and x^0 with respect to standard variables ξ^p, ξ^0 that accord with the weight functions from Table 1. Usually, one has $\xi^p \equiv \lambda^p$ and $\xi^0 \equiv \lambda^0$. For the sake of completeness, we consider here the case where λ^p and λ^0 do not necessarily belong in Table 1 and are thus expressed via $\xi^0 \in \mathbb{R}^{d_0}$ and $\xi^p \in \mathbb{R}^{d_p}$. In this way, the gPC representation of our state will depend on the concatenated random vector $\xi := ((\xi^p)^\top, (\xi^0)^\top)^\top \in \mathbb{R}^d$, with $d := d_p + d_0$. Specifically, one can write

$$\lambda^p_g \approx \sum_{j=0}^{\Lambda} \lambda^p_{gj}\phi^p_j(\xi^p), \quad x_i(t_0) \approx \sum_{j=0}^{K_0} x_{ij}(t_0)\phi^0_j(\xi^0), \quad x_i(t) \approx \sum_{j=0}^{K} x_{ij}(t)\phi_j(\xi) \tag{9}$$

for $g = 1, \ldots, d_p$ and $i = 1, \ldots, n$. Note that each set of polynomials is orthogonal with respect to its corresponding weight function. After plugging (9) into (8) and performing Galerkin projection (see (2)), one obtains

$$\dot{x}_{ij}(t) = \frac{\int_{\mathcal{D}} f_i(x(t,\xi), u(t), t, \xi)\phi_j(\xi)\rho(\xi)\mathrm{d}\xi}{\langle \phi_j, \phi_j\rangle}, \tag{10}$$

where $\rho(\cdot)$, $\mathcal{D}$ denote the density function and domain of ξ, respectively. Thus, the evolution of the gPC coefficients is compactly written as

$$\dot{\mathbf{X}}(t) = \mathbf{f}(\mathbf{X}(t), u(t), t), \tag{11}$$

with $\mathbf{f}$ described by (10) and $\mathbf{X} := (x_{10}, \ldots, x_{1K}, \ldots, x_{ij}, \ldots, x_{nK})^\top \in \mathbb{R}^{n(K+1)}$.

Polynomial chaos theory allows us to estimate the moments of a stochastic process analytically [35]. More precisely, using the orthogonality of $\{\phi_j\}$ we get

$$\begin{gathered}(\hat{\mathcal{M}}_i)_1(t) = x_{i0}(t), \quad (\hat{\mathcal{M}}_i)_2(t) = \sum_{j=1}^{K} x^2_{ij}(t)\langle \phi_j, \phi_j\rangle, \\ (\hat{\mathcal{M}}_i)_3(t) = \sum_{j=0}^{K}\sum_{g=0}^{K} x_{ij}(t)x_{ig}(t)\left(\sum_{l=0}^{K} x_{il}(t)\langle \phi_j, \phi_g, \phi_l\rangle - 3x_{i0}(t)\right) + 2x^3_{i0}(t), \quad \ldots .\end{gathered} \tag{12}$$

Here, $(\hat{\mathcal{M}}_i)_j$ denotes the gPC estimate for the jth (central) moment of state x_i. Consequently, propagating the ODEs from (11) gives an estimate about the state distribution over time. This further implies that the probabilistic evolution of our system can be influenced by controlling (11).

4.2. Discrete Euler–Lagrange equations for gPC expansions of mechanical systems. Faithful discrete representations of continuous dynamics play a key role in discrete-time control. Here, one could discretize (11) directly. However, such a naive approach will usually induce numerical errors during simulation.

Notice that any mechanical system will satisfy (4). In the presence of uncertain parameters $\xi \in \mathbb{R}^d$, one can use a gPC expansion on the coordinates q, v, and p. That is,

$$q_i(t) \approx \sum_{j=0}^{K} q_{ij}(t)\phi_j(\xi), \quad v_i(t) \approx \sum_{j=0}^{K} v_{ij}(t)\phi_j(\xi), \quad p_i(t) \approx \sum_{j=0}^{K} p_{ij}(t)\phi_j(\xi). \tag{13}$$

Now, define the concatenated vectors $\mathbf{Q} := (q_{10}, \ldots, q_{1K}, \ldots, q_{NK}) \in \mathbb{R}^{N(K+1)}$ and $\mathbf{V} := (v_{10}, \ldots, v_{1K}, \ldots, v_{NK}) \in \mathbb{R}^{N(K+1)}$. Let also $\hat{\mathbf{P}} := (\hat{p}_{10}, \ldots, \hat{p}_{1K}, \ldots, \hat{p}_{NK}) \in$

$\mathbb{R}^{N(K+1)}$ be the set of unnormalized momentum coefficients, with $\{\hat{p}_{ij}\}:=\{p_{ij}\langle\phi_j,\phi_j\rangle\}$. The authors in [27] showed that the polynomial chaos coefficients of uncertain conservative systems satisfy Hamilton's equations. Here, we extend this concept and determine the corresponding DEL equations, especially when nonconservative forces are applied. The main result is given by the following lemma.

LEMMA 4.1. *Consider a mechanical system with uncertain parameters ξ, and let L, F denote its Lagrangian function and set of nonconservative forces, respectively. Suppose also that its position, velocity, and momentum coordinates can be expanded as in* (13)*. Then the system of gPC coefficients* $(\mathbf{Q},\mathbf{V},\hat{\mathbf{P}})$ *will satisfy* (4)*, with*

$$\hat{L}(\mathbf{Q},\mathbf{V}) := \int_{\mathcal{D}} L\rho(\xi)\mathrm{d}\xi \tag{14}$$

being the associated Lagrangian function, and

$$\hat{\mathbf{F}} := (\hat{F}_{10},\dots,\hat{F}_{1K},\dots,\hat{F}_{NK}) \in \mathbb{R}^{N(K+1)}, \quad \hat{F}_{ij}(\mathbf{Q},\mathbf{V},u) := \int_{\mathcal{D}} F_i\phi_j(\xi)\rho(\xi)\mathrm{d}\xi \tag{15}$$

being the nonconservative forces.

Proof. For Lagrangian systems, (4) must be satisfied. Plugging (13) into (4) and performing Galerkin projection gives

$$v_{ij} = \dot{q}_{ij}, \tag{16}$$

$$\hat{p}_{ij} = \int_{\mathcal{D}} \frac{\partial L}{\partial v_i}\phi_j(\xi)\rho(\xi)\mathrm{d}\xi, \tag{17}$$

$$\dot{\hat{p}}_{ij} = \int_{\mathcal{D}} \frac{\partial L}{\partial q_i}\phi_j(\xi)\rho(\xi)\mathrm{d}\xi + \int_{\mathcal{D}} F_i\phi_j(\xi)\rho(\xi)\mathrm{d}\xi \tag{18}$$

for $i = 1,\dots,N$ and $j = 0,\dots,K$. Now, define the functions $\hat{L}$, $\hat{F}_{ij}$ as in (14) and (15), respectively. Since the gPC coefficients are deterministic, one can obtain

$$\frac{\partial \hat{L}}{\partial v_{ij}} = \int_{\mathcal{D}} \frac{\partial L}{\partial v_i}\frac{\partial v_i}{\partial v_{ij}}\rho(\xi)\mathrm{d}\xi = \int_{\mathcal{D}} \frac{\partial L}{\partial v_i}\phi_j(\xi)\rho(\xi)\mathrm{d}\xi, \tag{19}$$

where the first equality is due to the chain rule, and the second equality is due to (13). Equations (17) and (19) yield

$$\hat{p}_{ij} = \frac{\partial \hat{L}}{\partial v_{ij}}. \tag{20}$$

In a similar manner, one can show that

$$\frac{\partial \hat{L}}{\partial q_{ij}} = \int_{\mathcal{D}} \frac{\partial L}{\partial q_i}\frac{\partial q_i}{\partial q_{ij}}\rho(\xi)\mathrm{d}\xi = \int_{\mathcal{D}} \frac{\partial L}{\partial q_i}\phi_j(\xi)\rho(\xi)\mathrm{d}\xi. \tag{21}$$

Combining (15), (18), and (21) gives

$$\dot{\hat{p}}_{ij} = \frac{\partial \hat{L}}{\partial q_{ij}} + \hat{F}_{ij}. \tag{22}$$

The conclusion is made by comparing (16), (20), and (22) with (4). □

Lemma 4.1 implies that the set of gPC-based coordinates from (13) behaves as a Lagrangian system. In this context, $\mathbf{Q}$, $\mathbf{V}$, and $\hat{\mathbf{P}}$ denote the position, velocity, and (unnormalized) momentum coordinates, respectively. $\hat{L}$ is determined as the expectation of the original Lagrangian function, while each $\hat{F}_{ij}$ is given by a projection of the force F_i along the corresponding basis function.

Since Hamiltonian and Lagrangian principles are equivalent, the associated Euler–Lagrange equations will be satisfied in continuous time. As a consequence, one can define the DEL equations for the gPC representation as follows:

$$0 = \hat{p}_{ij}^k + \frac{\partial}{\partial q_{ij}^k}\hat{L}_d(\mathbf{Q}^k, \mathbf{Q}^{k+1}) + \hat{F}_{dij}^-(\mathbf{Q}^k, \mathbf{Q}^{k+1}, u^k), \tag{23a}$$

$$\hat{p}_{ij}^{k+1} = \frac{\partial}{\partial q_{ij}^{k+1}}\hat{L}_d(\mathbf{Q}^k, \mathbf{Q}^{k+1}) + \hat{F}_{dij}^+(\mathbf{Q}^k, \mathbf{Q}^{k+1}, u^k), \tag{23b}$$

where $i = 1, \ldots, N$ and $j = 0, \ldots, K$. When the midpoint rule is used, the discrete versions of the Lagrangian, $\hat{L}_d$, and nonconservative, $\hat{F}_{dij}^{\pm}$, forces can be computed as in (5) and (6), respectively. The required steps for propagating the discrete gPC representation of dynamical systems are presented in Algorithm 1.

Algorithm 1. VI for gPC representations of mechanical systems.

```
Data: Lagrangian $L$, nonconservative forces $F$, horizon $K_f$, $q^0$, and $p^0$;
Determine $\{(q_{ij}^0, \hat{p}_{ij}^0)\}$ by performing Galerkin projection on (13);
Compute $\hat{L}_d$ and $\{\hat{F}_{dij}^{\pm}\}$ by discretizing (14) and (15), respectively (e.g., using (5), (6));
for $k = 0$ to $K_f - 1$ do
    Given $\{(q_{ij}^k, \hat{p}_{ij}^k)\}$, compute $\{q_{ij}^{k+1}\}$ by implicitly solving (23a);
    Get $\{\hat{p}_{ij}^{k+1}\}$ by evaluating (23b);
end for
```

5. Numerical optimal control of stochastic systems.

We extend the original differential dynamic programming (DDP) algorithm to control polynomial chaos representations. Our approach relies on the observation that gPC theory typically augments the nominal state space. We will find that the scalability and fast convergence of our numerical scheme help mitigate this shortcoming. Details are also provided for incorporating the VI from subsection 4.2 when mechanical systems are considered. The section concludes by studying the convergence rate of our method, as well as the conditions for reaching a (sub)optimal solution.

5.1. gPC formulations of stochastic optimal control problems.

It is easy to see that (10)–(12) transform the moment-based problem (1) into its deterministic analogue:

$$\begin{aligned} &\min_u \int_{t_0}^{t_f} \boldsymbol{L}(\mathbf{X}, u, t)\mathrm{d}t + \boldsymbol{F}(\mathbf{X}(t_f), t_f) \\ &\text{s.t.}\quad \dot{\mathbf{X}}(t) = \mathbf{f}(\mathbf{X}, u, t), \quad \mathbf{X}(t_0) = \overline{\mathbf{X}}^0, \end{aligned} \tag{24}$$

where $\boldsymbol{L}(\mathbf{X}, u, t) := \mathcal{L}(\hat{\mathcal{M}}(\mathbf{X}(t)), u, t)$ and $\boldsymbol{F}(\mathbf{X}(t_f)) := \mathcal{F}(\hat{\mathcal{M}}(\mathbf{X}(t_f)), t_f)$. The augmented state is $\mathbf{X} := (x_{10}, \ldots, x_{1K}, \ldots, x_{ij}, \ldots, x_{nK})^\top \in \mathbb{R}^{\boldsymbol{n}}$, $\boldsymbol{n} := n(K+1)$, with the dynamic constraints given by (11). Moreover, since the statistics of the (uncertain) initial state, $x(t_0)$, are assumed to be known, $\mathbf{X}(t_0)$ will be fixed. Lastly, notice that (24) is an approximation of the original problem in (1); the two formulations become equivalent only when the orthogonal projections in (9) hold with equality.

As a concrete example, one can show that expected costs with quadratic terms (usually found in standard stochastic optimal control theory) can be viewed as a subclass of (24). The details are given in section SM1.

5.2. The gPC-DDP framework: Derivation in discrete time. In what follows, we will use superscripts to indicate evaluation at a particular time instant (e.g., $\mathbf{X}^k$ denotes the state at $t \equiv t_k$). Furthermore, our cost functions and dynamics are assumed to be twice differentiable. We begin by defining the value function:

$$V(\mathbf{X}, t) := \min_{u} \left[\int_{t}^{t_f} \boldsymbol{L}(\mathbf{X}, u, s)\mathrm{d}s + \boldsymbol{F}(\mathbf{X}(t_f), t_f) \right]. \tag{25}$$

From Bellman's principle of optimality in discrete time, we can write [16]

$$V^k(\mathbf{X}^k) = \min_{u^k} \left[\boldsymbol{L}^k(\mathbf{X}^k, u^k) + V^{k+1}(\mathbf{X}^{k+1}) \right], \tag{26}$$

where $\boldsymbol{L}^k(\mathbf{X}^k, u^k) := \boldsymbol{L}(\mathbf{X}(t_k), u(t_k))\Delta t$ is the discrete approximation of the corresponding time integral in (25). Here, the problem will be constrained by discrete (in time) transition dynamics of the form $\mathbf{X}^{k+1} = \mathbf{f}^k(\mathbf{X}^k, u^k)$. These can be derived from the continuous model in (24) either implicitly or explicitly (for example, the forward Euler method reads $\mathbf{f}^k = \mathbf{X}^k + \Delta t \mathbf{f}(\mathbf{X}^k, u^k, t^k)$).

Our goal is to find (sub)optimal controls $\{u^k\}$ that minimize the right-hand side of (26). To this end, both sides of (26) will be approximated via quadratic expansions. Let us first define the Q function as follows:

$$Q^k(\mathbf{X}^k, u^k) := \boldsymbol{L}^k(\mathbf{X}^k, u^k) + V^{k+1}(\mathbf{X}^{k+1}). \tag{27}$$

Next, given a nominal sequence $\{(\overline{\mathbf{X}}^k, \bar{u}^k)\}$, we will use a Taylor expansion to express the transition of state perturbations over time:

$$\begin{aligned} \delta\mathbf{X}^{k+1} \approx \nabla_{\mathbf{x}}\mathbf{f}^k \delta\mathbf{X}^k + \nabla_u \mathbf{f}^k \delta u^k + \frac{1}{2}\bigg(&\sum_{i,j=1}^{\boldsymbol{n}} \nabla_{\mathbf{x}_i\mathbf{x}_j}\mathbf{f}^k \delta\mathbf{X}_i^k \delta\mathbf{X}_j^k \\ &+ \sum_{i,j=1}^{m} \nabla_{u_i u_j}\mathbf{f}^k \delta u_i^k \delta u_j^k + 2\sum_{i,j=1}^{\mathbf{n},m} \nabla_{\mathbf{x}_i u_j}\mathbf{f}^k \delta\mathbf{X}_i^k \delta u_j^k \bigg), \end{aligned} \tag{28}$$

where $\delta\mathbf{X}^k := \mathbf{X}^k - \overline{\mathbf{X}}^k$ and $\delta u^k := u^k - \bar{u}^k$ correspond to state and control deviations, respectively. By using (28), we can apply a quadratic expansion on the right-hand side of (27). It is straightforward to obtain the expression

$$\begin{aligned} Q^k \approx Q_0^k &+ (\delta\mathbf{X}^k)^\top Q_{\mathbf{x}}^k + (\delta u^k)^\top Q_u^k + \tfrac{1}{2}\big((\delta\mathbf{X}^k)^\top Q_{\mathbf{xx}}^k \delta\mathbf{X}^k \\ &+ (\delta u^k)^\top Q_{uu}^k \delta u^k + (\delta u^k)^\top Q_{u\mathbf{x}}^k \delta\mathbf{X}^k + (\delta\mathbf{X}^k)^\top Q_{\mathbf{x}u}^k \delta u^k\big), \end{aligned} \tag{29}$$

with

$$\begin{aligned}
&Q_0^k := \boldsymbol{L}^k + V^{k+1}, \quad Q_{\mathbf{x}}^k := \boldsymbol{L}_{\mathbf{x}}^k + (\nabla_{\mathbf{x}}\mathbf{f}^k)^\top V_{\mathbf{x}}^{k+1}, \quad Q_u^k := \boldsymbol{L}_u^k + (\nabla_u\mathbf{f}^k)^\top V_{\mathbf{x}}^{k+1},\\
&Q_{\mathbf{xx}}^k := \boldsymbol{L}_{\mathbf{xx}}^k + (\nabla_{\mathbf{x}}\mathbf{f}^k)^\top V_{\mathbf{xx}}^{k+1}\nabla_{\mathbf{x}}\mathbf{f}^k + \sum_{i=1}^{\boldsymbol{n}}[V_{\mathbf{x}}^{k+1}]_i\nabla_{\mathbf{xx}}[\mathbf{f}^k]_i,\\
&Q_{\mathbf{x}u}^k := \boldsymbol{L}_{\mathbf{x}u}^k + (\nabla_{\mathbf{x}}\mathbf{f}^k)^\top V_{\mathbf{xx}}^{k+1}\nabla_u\mathbf{f}^k + \sum_{i=1}^{\boldsymbol{n}}[V_{\mathbf{x}}^{k+1}]_i\nabla_{\mathbf{x}u}[\mathbf{f}^k]_i, \quad Q_{u\mathbf{x}}^k := (Q_{\mathbf{x}u}^k)^\top,\\
&Q_{uu}^k := \boldsymbol{L}_{uu}^k + (\nabla_u\mathbf{f}^k)^\top V_{\mathbf{xx}}^{k+1}\nabla_u\mathbf{f}^k + \sum_{i=1}^{\boldsymbol{n}}[V_{\mathbf{x}}^{k+1}]_i\nabla_{uu}[\mathbf{f}^k]_i.
\end{aligned} \tag{30}$$

Here, $[\mathsf{g}]_i$ denotes the ith element of a vector g (also written as g_i). It is implied that all quantities in (30) are evaluated on the nominal sequence.

Now, observe that we can write our control input as $u^k = \bar{u}^k + \delta u^k$ for each instance k. $\{\bar{u}^k\}$ denotes a nominal control sequence and will be treated as a fixed quantity (in our algorithm, this will correspond to controls of the previous iteration). Based on this observation, we will optimize over control "perturbations" $\{\delta u^k\}$. Specifically, recalling (26) and (27), we can explicitly find control updates δu_*^k that minimize the approximation in (29). A straightforward manipulation gives

$$\delta u_*^k = -(Q_{uu}^k)^{-1}Q_u^k - (Q_{uu}^k)^{-1}Q_{u\mathbf{x}}^k\delta\mathbf{X}^k. \tag{31}$$

Note that the solution above has been derived on the basis of quadratic approximations of the cost and dynamics, and thus we cannot expect (31) to be valid for generic problems. To this end, we will employ a line search parameter $\gamma \in (0, 1]$ and compute the updated control inputs in an iterative fashion:

$$u_{(l+1)}^k = u_{(l)}^k - \gamma(Q_{uu,(l)}^k)^{-1}Q_{u,(l)}^k - (Q_{uu,(l)}^k)^{-1}Q_{u\mathbf{x},(l)}^k\delta\mathbf{X}_{(l)}^k. \tag{32}$$

Here, subscript (l) is associated with the lth iteration of our scheme, at which the nominal controls correspond to $\bar{u}^k \equiv u_{(l-1)}^k$; that is, the value of the decision variables at the previous iteration. Similarly, $\overline{\mathbf{X}}^k \equiv \mathbf{X}_{(l-1)}^k$ is the nominal trajectory at iteration (l), which is generated under controls $u_{(l-1)}^k$. We prove in subsection 5.4 and the supplementary material that our optimization algorithm will converge under (32) when a sufficiently small γ is used.

The reader may notice that γ affects only the feedforward term of $\delta u_{(l)}^k$ directly. To explain this choice, we point to section SM3, where it is shown that $\delta\mathbf{X}_{(l)}^k := \mathbf{X}_{(l+1)}^k - \mathbf{X}_{(l)}^k$ is already of order $O(\gamma)$ for each k. This, among other facts, allows us to establish convergence when the update rule in (32) is utilized. Potentially, one could have γ multiply the feedback term of $\delta u_{(l)}^k$ as well; however, we have not observed any associated benefit in theory or implementation under this approach, and we will therefore apply the equation above.

To proceed, computation of the new controls (32) requires knowledge of $V_{\mathbf{x}}^k$ and $V_{\mathbf{xx}}^k$ on the nominal trajectory. In light of this, $V^k(\mathbf{X}^k)$ from (26) will be expanded as

$$V^k(\mathbf{X}_{(l+1)}^k) \approx V^k|_{\mathbf{x}_{(l)}} + (\delta\mathbf{X}_{(l)}^k)^\top V_{\mathbf{x}}^k|_{\mathbf{x}_{(l)}} + \tfrac{1}{2}(\delta\mathbf{X}_{(l)}^k)^\top V_{\mathbf{xx}}^k|_{\mathbf{x}_{(l)}}\delta\mathbf{X}_{(l)}^k. \tag{33}$$

We will now plug expressions (27), (29), (32), and (33) into (26). This will allow us to match equal-order terms between Q^k and V^k, and thus obtain a recursive method for approximating $V^k_{\mathbf{x}}|_{\mathbf{x}_{(l)}}$ and $V^k_{\mathbf{xx}}|_{\mathbf{x}_{(l)}}$. Simple manipulations will give

$$\begin{aligned} V^k_{\mathbf{x}}|_{\mathbf{x}_{(l)}} &= Q^k_{\mathbf{x},(l)} - Q^k_{\mathbf{x}u,(l)}(Q^k_{uu,(l)})^{-1}Q^k_{u,(l)}, \\ V^k_{\mathbf{xx}}|_{\mathbf{x}_{(l)}} &= Q^k_{\mathbf{xx},(l)} - Q^k_{\mathbf{x}u,(l)}(Q^k_{uu,(l)})^{-1}Q^k_{u\mathbf{x},(l)}. \end{aligned} \tag{34}$$

Since Q^k depends on V^{k+1} (see (27), (30)), the equations above must be solved backwards in time. The boundary conditions will be given by the final cost term, that is, $V_{\mathbf{x}}(\mathbf{X}(t_f)) = \boldsymbol{F}_{\mathbf{x}}(\mathbf{X}(t_f))|_{\bar{\mathbf{x}}}$ and $V_{\mathbf{xx}}(\mathbf{X}(t_f)) = \boldsymbol{F}_{\mathbf{xx}}(\mathbf{X}(t_f))|_{\bar{\mathbf{x}}}$.

Before concluding this subsection, we note that gPC-DDP requires the Jacobians and Hessians of the polynomial chaos–based dynamics, $\mathbf{f}^k$. If a direct discretization of (11) is used, these will depend on the derivatives of $\mathbf{f}(\mathbf{X}, u, t)$. We show in the next proposition how to compute $\nabla_{\mathbf{x}}\mathbf{f}$ and $\nabla_{\mathbf{xx}}[\mathbf{f}]_\ell$ (expressions for the remaining derivatives can be obtained similarly and are thus omitted).

PROPOSITION 5.1. *Consider the gPC-based dynamics* $\mathbf{f}$ *given in* (11), *and let* $\dot{x}_{ij}$, *defined in* (10), *be its* ℓ*th element (i.e.,* $[\mathbf{f}]_\ell = \dot{x}_{ij}$*). Then*

$$\nabla_{\mathbf{x}}\mathbf{f} = \left[\int_{\mathcal{D}} \big(\nabla_x f(\xi) \otimes \big(\Phi(\xi) \otimes \Phi(\xi)^\top\big)\big)\rho(\xi)\mathrm{d}\xi\right] \odot \big(\mathbf{1}_{n\times n} \otimes \psi\big), \tag{35}$$

$$\nabla_{\mathbf{xx}}[\mathbf{f}]_\ell = \int_{\mathcal{D}} \left(\frac{\nabla_{xx}[f]_i(\xi)}{\langle \phi_j, \phi_j\rangle}\phi_j(\xi) \otimes \big(\Phi(\xi) \otimes \Phi(\xi)^\top\big)\right)\rho(\xi)\mathrm{d}\xi, \tag{36}$$

where $\otimes$ *is the Kronecker product,* $\odot$ *is the Hadamard product, and* $\mathbf{1}$ *is the unit matrix. Furthermore,* $\Phi \in \mathbb{R}^{K+1}$ *includes the* $K+1$ *orthogonal polynomials of the state vector (i.e.,* $\Phi = (\phi_0(\xi), \dots, \phi_K(\xi))^\top$*), while* $\psi = (1/\langle\phi_0,\phi_0\rangle, \dots, 1/\langle\phi_K,\phi_K\rangle)^\top \in \mathbb{R}^{K+1}$.

Proof. Consider the expression given in (10). Since the gPC coefficients are deterministic, one has

$$\frac{\partial \mathbf{f}_\ell}{\partial x_{gh}} = \frac{\int_{\mathcal{D}} \frac{\partial f_i(\xi)}{\partial x_{gh}}\phi_j(\xi)\rho(\xi)\mathrm{d}\xi}{\langle\phi_j,\phi_j\rangle} = \frac{\int_{\mathcal{D}} \frac{\partial f_i(\xi)}{\partial x_g}\frac{\partial x_g}{\partial x_{gh}}\phi_j(\xi)\rho(\xi)\mathrm{d}\xi}{\langle\phi_j,\phi_j\rangle} = \frac{\int_{\mathcal{D}} \frac{\partial f_i(\xi)}{\partial x_g}\phi_h(\xi)\phi_j(\xi)\rho(\xi)\mathrm{d}\xi}{\langle\phi_j,\phi_j\rangle},$$

where the second equality is due to chain rule, and the last one is due to (9). Similarly, it can be shown that $\frac{\partial^2 \mathbf{f}_\ell}{\partial x_{gh}\partial x_{sd}} = \frac{\int_{\mathcal{D}} \frac{\partial^2 f_i(\xi)}{\partial x_g \partial x_s}\phi_d(\xi)\phi_h(\xi)\phi_j(\xi)\rho(\xi)\mathrm{d}\xi}{\langle\phi_j,\phi_j\rangle}$ for $i, g, s = 1, \dots, n$ and $j, h, d = 0, \dots, K$. Based on these expressions, the result is obtained by writing $\nabla_{\mathbf{x}}\mathbf{f}$ and $\nabla_{\mathbf{xx}}[\mathbf{f}]_\ell$ in matrix form. □

Next, we address similar computations when a VI is employed.

5.2.1. Incorporating a variational integrator.

When we work with mechanical systems, we can use a VI for the propagation and linearization phases of gPC-DDP. Here, we will make use of the DEL equations in (23); as mentioned earlier, more involved schemes can be developed through Lemma 4.1.

Since $\mathbf{f}^k$ is determined implicitly from (23), we can use Algorithm 1 to get our nominal trajectories. Thus, it only remains to get an approximation analogous to (28). Towards this goal, we use the results from [17], where a linearization method for discrete systems was studied. For our case, this can be accomplished by differentiating (23a) implicitly with respect to $\mathbf{Q}^{k+1}$, and (23b) explicitly with respect to $\hat{\mathbf{P}}^{k+1}$. Due to space limitations we will only provide the final expressions. Let us define for compactness $\hat{L}^k := \hat{L}_d(\mathbf{Q}^k, \mathbf{Q}^{k+1})$ and $\hat{\mathbf{F}}^{k\pm} := \hat{\mathbf{F}}^{\pm}_d(\mathbf{Q}^k, \mathbf{Q}^{k+1}, u^k)$. Let also $\mathcal{X}^k :=$

$((\mathbf{Q}^k)^\top, (\hat{\mathbf{P}}^k)^\top)^\top \in \mathbb{R}^{\boldsymbol{n}}$ be our coordinate-based state vector. Then

$$\begin{aligned}\delta\mathcal{X}^{k+1} \approx \frac{\partial\mathcal{X}^{k+1}}{\partial\mathcal{X}^k}\delta\mathcal{X}^k + \frac{\partial\mathcal{X}^{k+1}}{\partial u^k}\delta u^k + \frac{1}{2}\Bigg(\sum_{i,j=1}^{\boldsymbol{n}} \frac{\mathcal{X}^{k+1}}{\partial\mathcal{X}_i^k\partial\mathcal{X}_j^k}\delta\mathcal{X}_i^k\delta\mathcal{X}_j^k \\ + \sum_{i,j=1}^{m}\frac{\mathcal{X}^{k+1}}{\partial u_i^k\partial u_j^k}\delta u_i^k\delta u_j^k + 2\sum_{i,j=1}^{\mathbf{n},m}\frac{\mathcal{X}^{k+1}}{\partial\mathcal{X}_i^k\partial u_j^k}\delta\mathcal{X}_i^k\delta u_j^k\Bigg).\end{aligned} \tag{37}$$

The first-order terms are determined as follows:

$$\frac{\partial\mathbf{Q}^{k+1}}{\partial\mathbf{Q}^k} = -(\hat{M}^k)^{-1}[D_1D_1\hat{L}^k + D_1\hat{\mathbf{F}}^{k-}], \quad \frac{\partial\mathbf{Q}^{k+1}}{\partial\hat{\mathbf{P}}^k} = -(\hat{M}^k)^{-1},$$
$$\frac{\partial\hat{\mathbf{P}}^{k+1}}{\partial\hat{\mathbf{P}}^k} = [D_2D_2\hat{L}^k + D_2\hat{\mathbf{F}}^{k+}]\frac{\partial\mathbf{Q}^{k+1}}{\partial\hat{\mathbf{P}}^k}, \quad \frac{\partial\hat{\mathbf{P}}^{k+1}}{\partial u^k} = [D_2D_2\hat{L}^k + D_2\hat{\mathbf{F}}^{k+}]\frac{\partial\mathbf{Q}^{k+1}}{\partial u^k} + D_3\hat{\mathbf{F}}^{k+},$$
$$\frac{\partial\mathbf{Q}^{k+1}}{\partial u^k} = -(\hat{M}^k)^{-1}D_3\hat{\mathbf{F}}^{k-}, \ \frac{\partial\hat{\mathbf{P}}^{k+1}}{\partial\mathbf{Q}^k} = [D_2D_2\hat{L}^k + D_2\hat{\mathbf{F}}^{k+}]\frac{\partial\mathbf{Q}^{k+1}}{\partial\mathbf{Q}^k} + D_1D_2\hat{L}^k + D_1\hat{\mathbf{F}}^{k+},$$

with $\hat{M}^k := D_2D_1\hat{L}^k + D_2\hat{\mathbf{F}}^{k-}$. The second-order derivatives are given in section SM2. Note that the inversion of $\hat{M}^k$ is guaranteed by the implicit function theorem [17]. Now, gPC-DDP can be implemented by simply substituting the following in (30): $\mathbf{f}_{\mathbf{x}}^k \leftarrow \frac{\partial\mathcal{X}^{k+1}}{\partial\mathcal{X}^k}$, $\mathbf{f}_u^k \leftarrow \frac{\partial\mathcal{X}^{k+1}}{\partial u^k}$, $\nabla_{\mathbf{x}_iu_j}\mathbf{f}^k \leftarrow \frac{\partial\mathcal{X}^{k+1}}{\partial\mathcal{X}_i^k\partial u_j^k}$, $\nabla_{\mathbf{x}_i\mathbf{x}_j}\mathbf{f}^k \leftarrow \frac{\partial\mathcal{X}^{k+1}}{\partial\mathcal{X}_i^k\partial\mathcal{X}_j^k}$, and $\nabla_{u_iu_j}\mathbf{f}^k \leftarrow \frac{\partial\mathcal{X}^{k+1}}{\partial u_i^k\partial u_j^k}$.

Finally, note that each term in (37) relies on Jacobians and Hessians of $\hat{L}^k$ and $\hat{\mathbf{F}}^{k\pm}$. Below we give an expression for $D_1D_1\hat{L}^k$. The remaining quantities can be determined similarly.

Proposition 5.2. *Let* $\hat{L}^k$ *be defined as in* (14)*. Then*

$$D_1D_1\hat{L}^k = \int_{\mathcal{D}}\big(D_1D_1L_d(q^k, q^{k+1}, \xi) \otimes \big(\Phi(\xi)\otimes\Phi(\xi)^\top\big)\big)\rho(\xi)\mathrm{d}(\xi), \tag{38}$$

where Φ *has been defined in Proposition* 5.1.

Proof. First, observe from (5) and (14) that $\hat{L}^k = \int_{\mathcal{D}} L_d(q^k, q^{k+1}, \xi)\rho(\xi)\mathrm{d}\xi$. Due to (13), an approach similar to the proof of Proposition 5.1 will give the following: $\frac{\partial^2\hat{L}^k}{\partial q_{gh}^k\partial q_{ij}^k} = \int_{\mathcal{D}}\frac{\partial^2 L_d(q^k, q^{k+1}, \xi)}{\partial q_i^k\partial q_g^k}\phi_h(\xi)\phi_j(\xi)\rho(\xi)\mathrm{d}\xi$ for each $i, g = 1, \ldots, N$ and $j, h = 0, \ldots, K$. The result follows by writing $D_1D_1\hat{L}^k$ in matrix form. □

5.3. The gPC-DDP algorithm. We summarize the steps of gPC-DDP in Algorithm 2. Notice that the propagation and linearization phases require computing integrals of the form $\int_{\mathcal{D}}\mathsf{f}(\xi)\rho(\xi)\mathrm{d}\xi$, where $\mathsf{f} \in L_\rho^2$ (see, e.g., (10), (35), (36), and (38)). These quantities can rarely be calculated analytically, except when linear or sufficiently simple dynamics are considered. For our simulations (specifically, in subsection 6.2), we employed *Gaussian quadrature*. Given the density function $\rho(\xi)$, this method selects nodes $\{z_{l_i}\}$ and weights $\{w_{l_i}\}$ such that

$$\int_{\mathcal{D}_1}\cdots\int_{\mathcal{D}_d}\mathsf{f}(\xi)\rho(\xi)\mathrm{d}\xi \approx \sum_{l_1=1}^{l_{gq}}\cdots\sum_{l_d=1}^{l_{gq}} w_{l_1}\ldots w_{l_d}\mathsf{f}(z_{l_1}, \ldots, z_{l_d}), \quad \xi\in\mathbb{R}^d, \quad z_{l_i}\in\mathbb{R}. \tag{39}$$

Here, $\{z_{l_i}\}$ and $\{w_{l_i}\}$ are obtained by solving an eigenvalue decomposition problem [13]. The above formula holds with equality when the integrand is a polynomial of

Algorithm 2. gPC-DDP.

```
Data: Optimal control problem (24), initial nominal control sequence $\{u^k_{(0)}\}$;
Initialize iteration index, $l \leftarrow 0$;
Get the nominal state trajectory, $\{\mathbf{X}^k_{(l)}\}$, under $\{u^k_{(l)}\}$ by discretizing (11) (or using Algorithm 1);
repeat
    Starting from $t_f$, backpropagate the derivatives of $V$ and $Q$ along $\{(\mathbf{X}^k_{(l)}, u^k_{(l)})\}$ through (30), (34). This requires computing the linearization in (28) (or (37)) at each time step;
    Pick $\gamma = 1$. Starting from $t_0$ with $\delta\mathbf{X}^0_{(l)} = 0$, determine the new controls $u^0_{(l+1)}$ from (32), as well as the next corresponding state $\mathbf{X}^1_{(l+1)}$, and deviation $\delta\mathbf{X}^1_{(l)}$. Continue until the end of the horizon is reached. Accept the obtained trajectory if cost reduction is attained. Otherwise, reduce $\gamma \in (0,1]$ and repeat;
    Update $l \leftarrow l + 1$;
until convergence
```

degree less than or equal to $2l_{gq} - 1$. Since we consider smooth dynamics in this paper, the particular integration method is expected to be highly accurate. Lastly, observe that the full-tensor expression in (39) uses l_{gq}^d function evaluations. When d is large, one can use sparse quadrature formulas to reduce the number of nodes while retaining sufficient precision. More details can be found in [10].

Remark 5.1. gPC-DDP alternates between propagating nominal rollouts from the stochastic system and computing (sub)optimal control deviations about them. Hence, dynamics constraints are inherently satisfied by our optimizer. As shown in our simulations section, this is a major benefit, especially when augmented state-spaces under polynomial chaos expansions have to be handled.

5.4. Convergence analysis of gPC-DDP.

The analysis presented here goes along the lines of [21] and [22]. However, these works relied on certain simplifications to obtain their results. For instance, [21] did not consider running costs in its formulation, while [22] dealt only with scalar dynamic systems. In contrast, we will address the generic optimal control problem in (24).

In what follows, $U \in \mathbb{R}^{(K_f-1)m}$ denotes an entire control sequence (i.e., $U := ((u^0)^\top, (u^1)^\top, \dots, (u^{K_f-1})^\top)$). Also, we consider the next set of assumptions.

Assumption 5.1. (i) The dynamics and cost functions of (24) are differentiable up to the third order over a compact convex set $\mathcal{R} \in \mathbb{R}^{\boldsymbol{n}+m}$; (ii) $((\boldsymbol{X}^k)^\top, (u^k)^\top)^\top \in \mathcal{R}$ for all k; (iii) $Q^k_{uu} > 0$ for each $k = 0, 1, \dots$.

The convergence properties of gPC-DDP are established by the next theorems and corollaries. In particular, (41) implies that gPC-DDP will converge to a solution when the line-search parameter γ is sufficiently small. Furthermore, (42) points to a (locally) quadratic convergence rate of the algorithm.

THEOREM 5.1. *Consider the discrete-time optimal control problem*

$$\begin{aligned} &\min_{u^0, u^1, \dots, u^{K_f-1}} \quad \boldsymbol{J} \\ &s.t. \quad \mathbf{X}^{k+1} = \mathbf{f}^k(\mathbf{X}^k, u^k), \quad \mathbf{X}^0 = \overline{\mathbf{X}}^0, \end{aligned} \tag{40}$$

where $\boldsymbol{J} := \sum_{k=0}^{K_f-1} \boldsymbol{L}^k(\mathbf{X}^k, u^k) + \boldsymbol{F}(\mathbf{X}^{K_f})$. *Let also* $\delta U_* := ((\delta u^0_*)^\top, (\delta u^1_*)^\top, \dots)$ *in-*

clude the control updates of gPC-DDP, with $\delta u_*^k := -\gamma (Q_{uu}^k)^{-1} Q_u^k - (Q_{uu}^k)^{-1} Q_{ux}^k \delta \mathbf{X}^k$*. Then*

$$(\nabla_U \boldsymbol{J}|_{\bar{U}})^\top \delta U_* = -\gamma \sum_{k=0}^{K_f-1} (Q_u^k)^\top (Q_{uu}^k)^{-1} Q_u^k + O(\gamma^2). \tag{41}$$

Proof. See section SM3. □

Corollary 5.1. *Suppose Assumption* 5.1 *holds. Then, given any bounded* $\mathbf{X}^0$*, the gPC-DDP algorithm will converge to a stationary solution of* (40)*.*

Proof. The proof is in section SM4. □

Theorem 5.2. *Suppose Assumption* 5.1 *holds. Let also* $U_{(l)}$ *denote the controls obtained at iteration* l*, and let* $U_\star$ *be the stationary point gPC-DDP converges to. Then there exist* $c > 0$ *and* $l_\star > 0$*, such that*

$$||U_{(l)} - U_\star|| \leq c||U_{(l-1)} - U_\star||^2 \quad \forall l \geq l_\star. \tag{42}$$

Therefore, the convergence rate will be locally quadratic.

Proof. See section SM5. □

Remark 5.2. One can account for Assumption 5.1(iii) by setting [25]

$$Q_{uu}^k \leftarrow Q_{uu}^k + \theta I.$$

Here, $\theta > 0$ is a regularization parameter that enforces the required condition for all time instances.

6. Simulated examples.

First, a low-dimensional system is considered, namely, the Duffing oscillator. This will allow us to illustrate the algorithm in detail and make certain simplifications when employing the gPC expansions. Subsequently, we apply gPC-DDP on a quadrotor model to further elucidate the behavior of our framework.

6.1. Duffing oscillator.

The dynamics are given by

$$\dot{x} = \begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} x_2 \\ -\lambda x_1 - \frac{1}{4} x_2 - x_1^3 + u \end{pmatrix}. \tag{43}$$

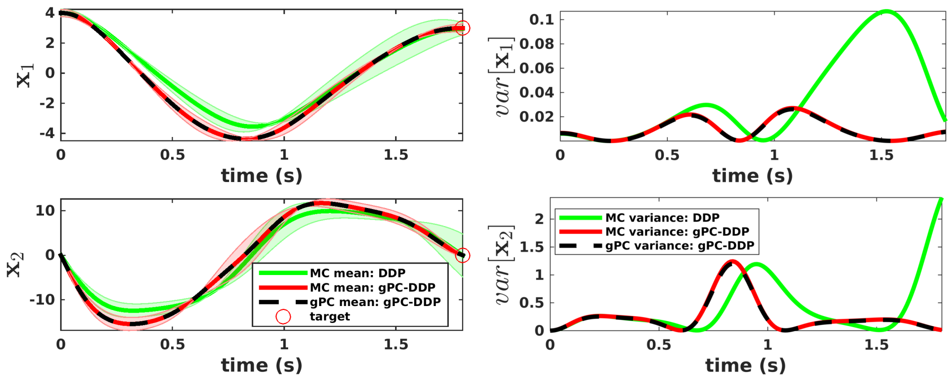
The uncertainty of our system will lie in the parameter λ, as well as the initial state $x_1(t_0)$. We consider these quantities to be normally distributed, with $\lambda \sim \mathcal{N}(\mu_\lambda, \sigma_\lambda^2)$ and $x_1(t_0) \sim \mathcal{N}(\mu_1^0, (\sigma_1^0)^2)$. Based on Table 1, Hermite polynomials will be employed, some expressions of which are given in subsection SM6.1. By applying (9), λ and $x_1(t_0)$ can be fully described as

$$\lambda(\xi^p) = \mu_\lambda \phi_0^p(\xi^p) + \sigma_\lambda \phi_1^p(\xi^p), \quad x_1^0(\xi^0) = \mu_1^0 \phi_0^0(\xi^0) + \sigma_1^0 \phi_1^0(\xi^0), \tag{44}$$

with $\xi^p, \xi^0 \sim \mathcal{N}(0,1)$. In this setting, the states are influenced by $\xi = (\xi^p, \xi^0) \in \mathbb{R}^2$. Applying the gPC expansion on the state vector yields

$$x_1(t,\xi) \approx \sum_{j=0}^{K} x_{1j}(t)\phi_j(\xi), \quad x_2(t,\xi) \approx \sum_{j=0}^{K} x_{2j}(t)\phi_j(\xi), \tag{45}$$

with $\phi_j(\xi) = \phi_{j_1}^p(\xi^p)\phi_{j_2}^0(\xi^0)$, $0 \leq j_1 + j_2 \leq r$ (r being the maximum total order of each ϕ_j). Since $x_2(t_0)$ is deterministic, we take $x_{20}(t_0) = x_2(t_0)$ and $x_{2j}(t_0) = 0$ for



(a) State trajectories (b) State variance

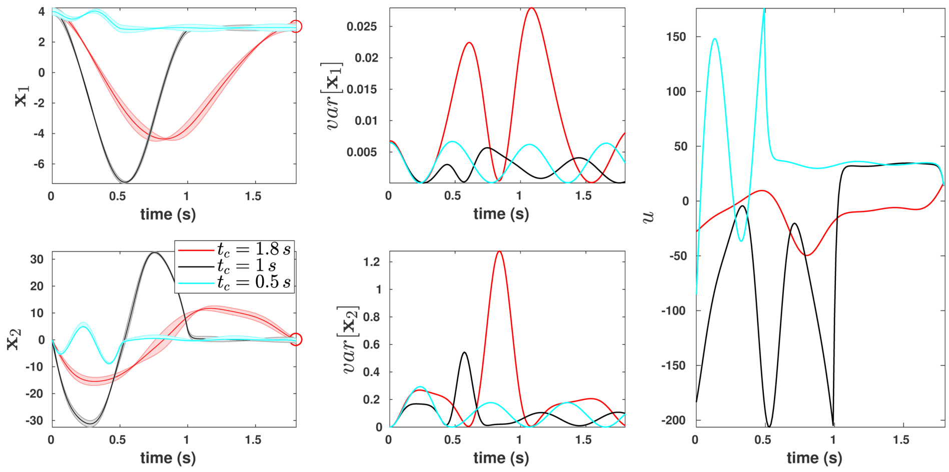
FIG. 1. *Duffing oscillator: Comparison between gPC-DDP and deterministic DDP (the latter applied on the mean parameter values). On the left-hand side, the Monte Carlo and gPC estimates of the expected states are illustrated, along with* $\pm 3\sigma$ *of sampled trajectories (green and red shaded areas). The right-hand side depicts the variance (gPC and MC estimates) of the state trajectories obtained by each algorithm. (Color available online.)*

$j > 0$. Plugging (44) and (45) into (43) and performing Galerkin projection as in (10) gives

$$
\begin{aligned}
&\dot{x}_{1l} = x_{2l},\\
&\dot{x}_{2l} = \frac{1}{\langle \phi_l, \phi_l \rangle}\Bigg(-\sum_{i=0}^{K} x_{1i}\big(\mu_\lambda \langle \phi_0^p, \phi_l, \phi_i \rangle + \sigma_\lambda \langle \phi_1^p, \phi_l, \phi_i \rangle\big) - \frac{1}{4} x_{2l} \langle \phi_l, \phi_l \rangle \\
&\qquad - \sum_{i=0}^{K}\sum_{g=0}^{K}\sum_{j=0}^{K} x_{1i}x_{1g}x_{1j} \langle \phi_l, \phi_i, \phi_g, \phi_j \rangle + \langle \phi_l \rangle u \Bigg), \quad l = 0, \dots, K.
\end{aligned} \tag{46}
$$

For our simulations we pick $\mu_\lambda = 3$, $\sigma_\lambda = 0.1$, $\mu_1^0 = 4$, $\sigma_1^0 = 0.08$, and $x_2^0 = 0$. Our task will be to reach the target state $x^{goal} = (3, 0)^\top$ in $t_f = 1.8$ s. The running cost is set to $\boldsymbol{L} = \frac{1}{2} 0.01 u^2$, while the terminal cost term is selected as $\boldsymbol{F} = \frac{1}{2}\sum_{i=1}^{2}((x_{i0}(t_f) - x_i^{goal})^2 s_{f_{i0}} + \sum_{l=1}^{K} x_{il}^2(t_f) s_{f_{ij}})$, with $s_{f_{10}} = s_{f_{20}} = 400$, and $s_{f_{1j}} = 300$, $s_{f_{2j}} = 100$ for each $j > 0$. Hence, gPC-DDP will penalize trajectories with (i) expected terminal states far from the desired ones, and (ii) high terminal variance. For the gPC expansions, we selected $r = 3$, resulting in an $\boldsymbol{n} = 20$ state vector (see (45)). Due to space limitations, we will not show how the accuracy of the gPC approximation changes with the number of expansion terms, K. In general, the estimated moments get closer to the actual ones as K increases, with the convergence rate depending on the quantity to be approximated (see discussion in section 3.1, as well as [36, 11, 35]). The results of our algorithm are displayed in Figures 1 and 3. Therein, we have used a forward Euler discretization of (46). Moreover, gPC-DDP was initialized with zero controls, while the convergence criterion was set to $J_{(i)} \leq 10^{-8}$ (here, $J_{(i)}$ is the cost at iteration i).

In Figure 1 we make a qualitative comparison between gPC-DDP and the original DDP method. The latter scheme was applied for the mean values of the uncertain parameters. Its terminal cost was defined as $\frac{1}{2}\sum_{i=1}^{2}((x_i(t_f) - x_i^{goal})^2 s_{f_{i0}}$, while its running cost was set equal to $\boldsymbol{L}$. Figure 1(a) shows the evolution of the gPC and Monte Carlo estimates of the mean, as well as $\pm 3\sigma$ of 1000 trajectories under the (locally)



(a) State trajectories (b) State variance (c) Controls

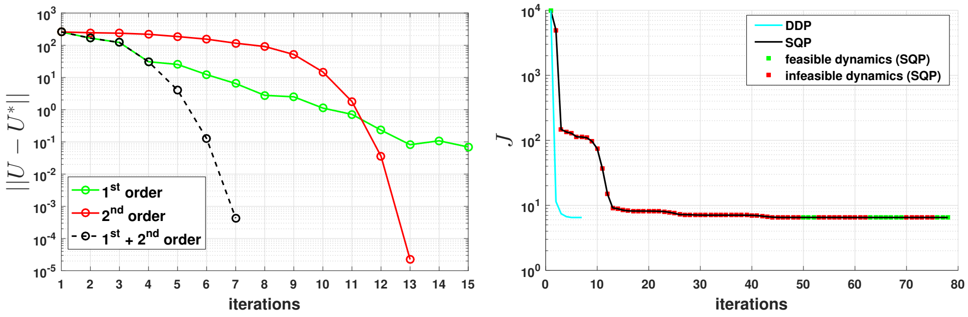
FIG. 2. *Duffing oscillator: Output of gPC-DDP when state-dependent running cost is used over the horizon $[t_c, t_f]$. Different colors correspond to different values of t_c. The details of the first two graphs are the same as in Figure* 1 *(gPC estimates of the mean and variance are omitted for clarity, since they are highly accurate in this example). The third plot shows the obtained controls for each case. (Color available online.)*

optimal controls. Furthermore, Figure 1(b) depicts the gPC and Monte Carlo variance estimates of the states under the different control settings. gPC-DDP is capable of guiding the expected states to the target while also minimizing the terminal variance.

Next, we show how the gPC-DDP algorithm behaves when state-dependent terms are added to the running cost. Specifically, we use $\boldsymbol{L}(\mathbf{X}(t), u(t)) = \frac{1}{2}0.01u(t)^2 + \frac{1}{2}\sum_{i=1}^{2}((x_{i0}(t) - x_i^{goal})^2 s_{f_{i0}} + \sum_{l=1}^{K} x_{il}^2(t) s_{f_{ij}})$ when $t \geq t_c$, and $\boldsymbol{L}(\mathbf{X}(t), u(t)) = \frac{1}{2}0.01u(t)^2$ otherwise (weight values are the same as before). In other words, uncertain states far from x^{goal} will now get penalized over the interval $t \in [t_c, t_f]$. Figure 2 shows the output of gPC-DDP for different time instances t_c. Therein, we see that trajectories are capable of staying close to x_{goal} over $[t_c, t_f]$, while keeping variance levels low. We also find that higher control inputs are usually required when t_c gets closer to t_0, as depicted in Figure 2(c).

In Figure 3(a) we provide further details about the behavior of gPC-DDP. Therein, we show the effect the linearization order has on the speed of convergence (plots correspond to control-dependent running cost). In particular, we compare using only the first-order terms from (28) as opposed to the full expansion. We observe that the former scheme typically yields improved results at the early stages of optimization. However, it fails to attain quadratic convergence rates. On the other hand, employing a second-order scheme achieves superlinear convergence as we move closer to the local solution, but is slower in the first iterations of gPC-DDP. In fact, we have noticed that the latter approach usually requires regularizing Q_{uu} more frequently (see Remark 5.2). These observations are supported by our analysis from subsection 5.4. Specifically, when proving convergence of the algorithm in Corollary 5.1, we only relied on a first-order linearization. However, when showing that gPC-DDP achieves locally quadratic convergence in Theorem 5.2, we made explicit use of the second-order terms from (28).

We propose switching between the two schemes, depending on the progress of



(a) gPC-DDP covergence rates

(b) DDP vs SQP for gPC-based systems

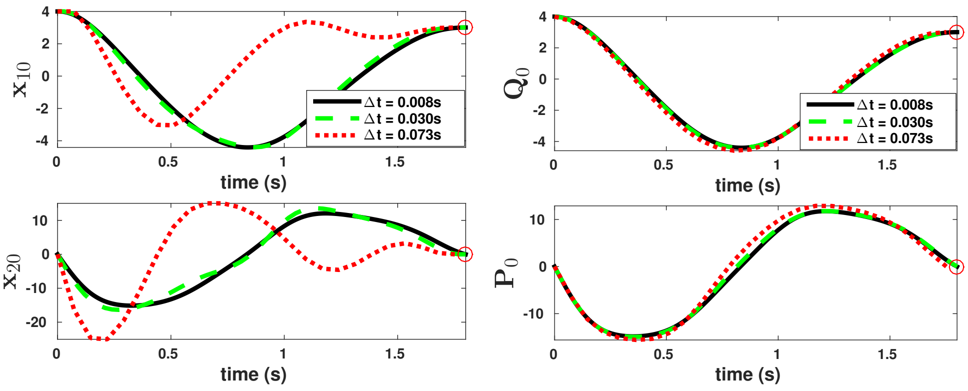
Fig. 3. *Duffing oscillator: The left figure shows how different expansion orders of dynamics affect the convergence rate of gPC-DDP. The green and red lines correspond to a first- and second-order approximation scheme, respectively. In contrast, the dashed black line employs the quadratic terms only from the fourth iteration, thus reaching the solution in fewer steps. The right figure compares gPC-DDP to an off-the-shelf SQP solver and depicts the cost value per iteration for each method. Both approaches reached the same solution. However, SQP was approximately* 70 *times slower and required more iterations to converge. Additionally, the black-box solver yielded infeasible dynamics for most iterations (the corresponding tolerance was set to* 10^{-7}*). (Color available online.)*

gPC-DDP. Specifically, we employ a first-order expansion during the first iterations and shift to a quadratic model when the change in the cost gets under a prespecified threshold, ΔJ^{th}. In this way, the higher-order terms will yield increased convergence rates in the neighborhood of $U_\star$, without degrading the early stages of our algorithm. This behavior is illustrated by the black dashed line in Figure 3(a). For our example, we picked $\Delta J^{th} = 10^{-3} J_{(1)}$, and the order switching occurred at the fourth iteration.

To proceed, Figure 3(b) includes the results of an off-the-shelf optimization solver as benchmark comparison. Therein, we employed the built-in MATLAB SQP implementation for solving problem (40). Derivative information about the cost and dynamics was also utilized to speed up convergence.

Using black-box direct algorithms for trajectory optimization is rather common (see, e.g., [8, 23, 20]). This is mainly due to the ease of implementation and simplicity of handling state constraints. Nonetheless, working in a large search space with many equality constraints can significantly degrade such an approach. Notice that when gPC expansions are employed, our search space becomes $\mathbb{R}^{(K_f n(K+1)+(K_f-1)m)}$, with n, m being the nominal state and controls dimension, respectively. In addition, $K_f n(K+1)$ equality constraints have to be satisfied throughout the time horizon. Based on these observations, it is intuitive why the off-the-shelf solver requires more iterations and time to converge. Moreover, since SQP tends to relax the constraints towards finding a better solution [25], we noticed that most of its intermediate steps yielded infeasible dynamics.

Utilizing a VI: Next, we incorporate the VI of subsection 4.2 into our scheme. The obtained control algorithm will be referred to as “VI-gPC-DDP.” Towards this goal, we need to determine the Lagrangian function and nonconservative forces for our gPC coefficients. Notice that the nominal Lagrangian and nonconservative forces of the Duffing oscillator are given respectively by $L(q,\dot{q}) = \frac{1}{2}\dot{q}^2 - \frac{\lambda}{2}q^2 - \frac{1}{4}q^4$ and $F(q,\dot{q},u) = u - \frac{1}{4}\dot{q}$. By expanding the generalized coordinates as $q \approx \sum_{j=0}^{K} q_j \phi_j(\xi)$,



(a) gPC-DDP (b) VI-gPC-DDP

FIG. 4. *Duffing oscillator: Comparison between gPC-DDP and VI-gPC-DDP for different step sizes, Δt. The former method used an explicit Euler scheme to propagate and linearize the gPC-based dynamics. The gPC mean estimates are able to reach the target for all cases. However, VI-gPC-DDP is much more insensitive to the selection of Δt. (Color available online.)*

$\dot{q} \approx \sum_{j=0}^{K} \dot{q}_j \phi_j(\xi)$, we find from (14) and (15) that

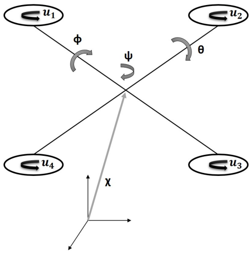
$$\begin{aligned}\hat{L} &= -\frac{1}{2}\sum_{j=0}^{K}\sum_{g=0}^{K} q_j q_g \big(\mu_\lambda \langle \phi_0^p, \phi_j, \phi_g\rangle + \sigma_\lambda \langle \phi_1^p, \phi_j, \phi_g\rangle\big)\\ &\quad + \frac{1}{2}\sum_{j=0}^{K} \dot{q}_j^2 \langle \phi_j, \phi_j\rangle - \frac{1}{4}\sum_{l=0}^{K}\sum_{h=0}^{K}\sum_{g=0}^{K}\sum_{j=0}^{K} q_l q_h q_g q_j \langle \phi_l, \phi_h, \phi_g, \phi_h\rangle,\\ \hat{F}_j &= u\langle \phi_j\rangle - \frac{1}{4}\dot{q}_j\langle \phi_j, \phi_j\rangle, \quad j = 0, \dots, K.\end{aligned}$$

For our implementation, the expressions above were discretized via (5) and (6).

To show the importance of our variational integration scheme, we compare the (locally) optimal trajectories given by gPC-DDP and VI-gPC-DDP for varying time steps, Δt. In general, a properly discretized model will behave similarly to its continuous counterpart, even for relatively large values of Δt. In that case, different time-step selections would only induce small deviations in the solutions of (40). Figure 4 shows the gPC mean estimates after implementing each of the aforementioned settings (plots correspond to control-dependent running cost; see also Figure 1). It can be easily deduced that utilizing our VI highly reduces the impact of Δt on the obtained trajectories.

6.2. Quadrotor. The state vector considered here is $x := (\chi^\top, \eta^\top, \dot{\chi}^\top, \dot{\eta}^\top)^\top \in \mathbb{R}^{12}$, where $\chi := (\mathsf{x}, \mathsf{y}, \mathsf{z})^\top \in \mathbb{R}^3$, $\eta := (\phi, \theta, \psi)^\top \in \mathbb{R}^3$ denote its position and Euler angles, respectively (see Figure 5(a) for an illustration). Details about the dynamics of this system are given in section SM7. Regarding its internal parameters (see Figures 5(b) and 5(c) for values thereof), l is the distance between the rotors and center of mass, m is the mass, g is the gravitational acceleration, $\mathbb{I} := \mathrm{diag}(\mathbb{I}_\mathsf{x}, \mathbb{I}_\mathsf{y}, \mathbb{I}_\mathsf{z})$ is the inertia matrix, and G_{tr}, G_{rot} are constants whose values depend on the air density and rotor shape.

We consider the case where G_{tr} and G_{rot} are uniformly distributed, with known upper and lower bounds. Now, based on Table 1, we will employ Legendre polynomials



(a) Illustration

g	$9.81\ m/s^2$
m	$1\ kg$
l	$0.24\ m$
$\mathbb{I}_x$	$8.1 \cdot 10^{-3}\ kgm^2$
$\mathbb{I}_y$	$8.1 \cdot 10^{-3}\ kgm^2$
$\mathbb{I}_z$	$14.2 \cdot 10^{-3}\ kgm^2$

(b) Deterministic parameters

	min	max
G_{tr}	$2.85 \cdot 10^{-5}$	$2.95 \cdot 10^{-5}$
G_{rot}	$1.05 \cdot 10^{-6}$	$1.15 \cdot 10^{-6}$

(c) Uniformly distributed, stochastic parameters

FIG. 5. *The quadrotor model—details.*

for our gPC expansions (for some useful expressions refer to subsection SM6.2). We picked the maximum order of our polynomials to be $r = 2$. This resulted in an $\boldsymbol{n} = 72$ state vector. Note also that in this example the propagation and linearization phases of gPC-DDP cannot be computed analytically. Hence, we utilized Gaussian quadrature, for which we employed $l_{gq} = 3$ quadrature nodes (see (39)).

Similarly to the previous problem, the cost functions are defined as follows: $\boldsymbol{F} = \frac{1}{2}\sum_{i=1}^{12}((x_{i0}(t_f) - x_i^{goal})^2 s_{f_{i0}} + \sum_{l=1}^{K} x_{il}^2(t_f) s_{f_{ij}})$ and $\boldsymbol{L} = \frac{1}{2}0.1\sum_{i=1}^{4} u_i^2$. Here, the target state is $x^{goal} = \mathbf{0}_{12}$, while the deterministic initial state is set to $x^0 = (-3, 3, 3, \mathbf{0}_9)$. Moreover, gPC-DDP was implemented using an order-switching for its linearizations, as explained in subsection 6.1 and Figure 3(a).

The results are depicted in Figures 6 and 7. To illustrate the impact of uncertainty penalization on our solutions, we consider two settings with different—variance related—weights $s_{f_{ij}}$ $(j = 1, \ldots, K)$. Specifically, trajectories in red correspond to $s_{f_{ij}} = 0.0013$, while those in magenta correspond to $s_{f_{ij}} = 0.044$. The weights associated with the expected states have been set to $s_{f_{i0}} = 8$ for all cases. As before, we also make a qualitative comparison with the original DDP algorithm applied for the mean values of G_{tr} and G_{rot}. Lastly, an illustration of gPC-DDP's (sub)optimal expected trajectory is given in Figure 8 (the latter figure was generated with $s_{f_{ij}} = 0.0013$ as its variance-related weights).

We make the following observation: As expected, high uncertainty penalization leads to "tighter" state trajectories around the target. However, Figure 7 shows that this also causes our control inputs to fluctuate more frequently. In fact, this behavior was noticed in all of our simulations. A more extensive study is required to establish a connection between uncertainty minimization and control effort.

Utilizing a VI: The aforementioned plots were generated through an explicit Euler discretization method. Let us now test the effect of our VI on this high-dimensional problem. In particular, we consider the following scenario: We begin by solving the optimization problem for different time steps, Δt. Then we evaluate each solution by plugging the corresponding controls into the actual, continuous[1] dynamics (11). Figure 9 compares three of the gPC mean estimates between an Euler-based gPC-DDP approach and VI-gPC-DDP. It is deduced that naive discretizations can highly degrade the planning procedure, especially when relatively large time steps are employed.

Being able to use coarse discretizations can also reduce the computational complexity of our algorithm. Figure 10 compares VI-gPC-DDP to gPC-DDP in terms of

[1] Here, we simulate continuous-like dynamics by applying the forward Euler method on (11) with a very small discretization step ($\Delta t = 0.001$).

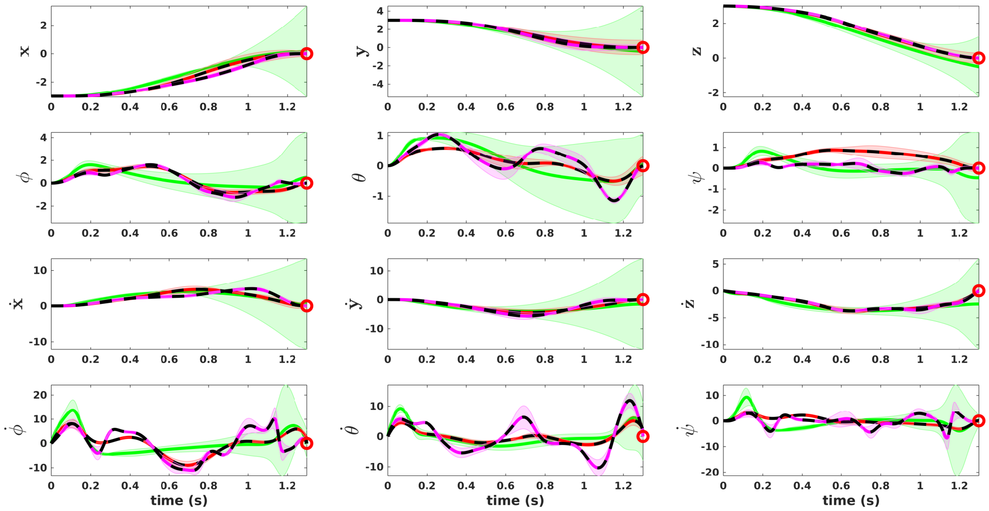


FIG. 6. *Quadrotor: Comparison between gPC-DDP (red and magenta) and deterministic DDP (green). Regarding the former, two settings are considered with different uncertainty penalization levels: red = low, magenta = high. Solid lines represent Monte Carlo mean estimates, while black dashed lines represent gPC mean estimates. The colored shaded areas correspond to* $\pm 3\sigma$ *of trajectories sampled under the different control sequences. Each algorithm was initialized with zero controls (units:* (x, y, z)*: m;* (ϕ, θ, ψ)*: rad;* $(\dot{x}, \dot{y}, \dot{z})$*: m/s;* $(\dot{\phi}, \dot{\theta}, \dot{\psi})$*: rad/s). (Color available online.)*

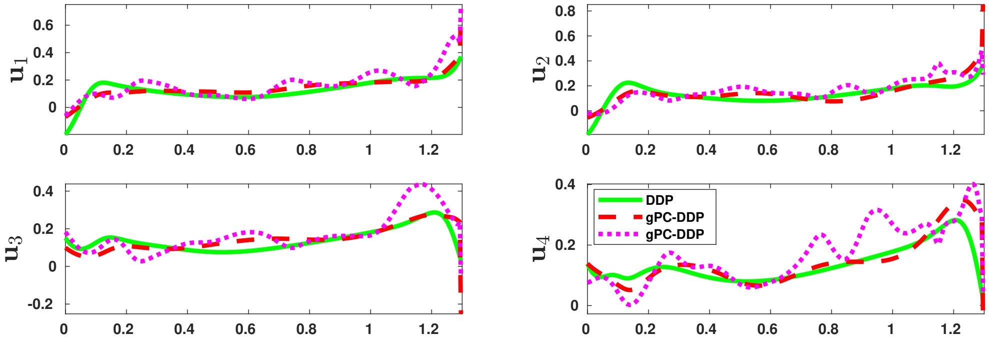


FIG. 7. *Quadrotor: Controls obtained by deterministic DDP (green) and gPC-DDP for different uncertainty penalization levels (red = low, magenta = high). (Color available online.)*

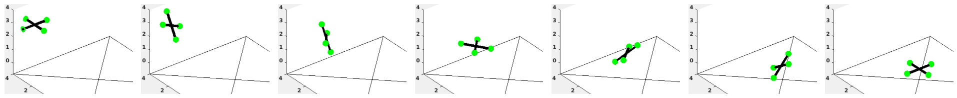
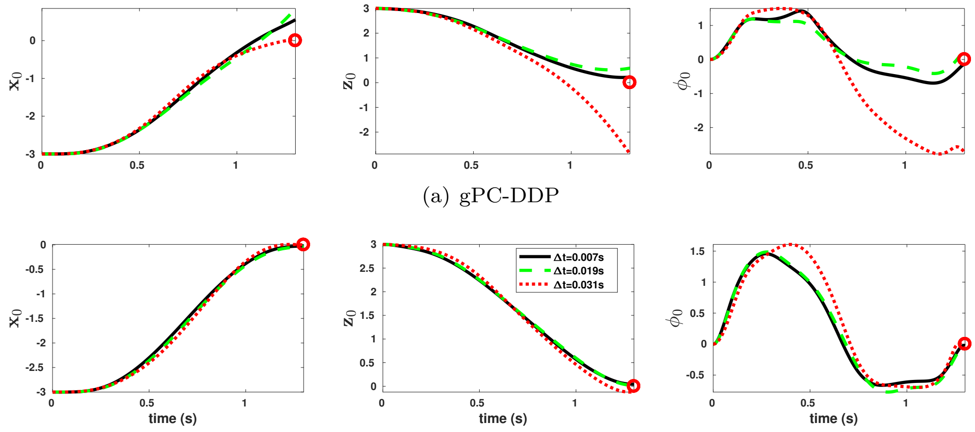


FIG. 8. *Quadrotor: Instances of the (sub)optimal, mean state trajectory obtained by gPC-DDP.*

the average time required for their propagation phases. Notice that the variational integration scheme will be slower for similar step sizes, since one has to solve (23b) implicitly. Nevertheless, when the discrete horizon is sufficiently decreased, the overall elapsed time will be reduced.

7. Discussion.

In this section, we mention possible extensions of our work that can further increase its applicability.



(a) gPC-DDP

(b) VI-gPC-DDP

FIG. 9. *Quadrotor: Comparison between gPC-DDP and VI-gPC-DDP. The legend shows the discretization step that we used when implementing each scheme. The plots were obtained by applying the (sub)optimal controllers on the continuous gPC-based dynamics. (Color available online.)*

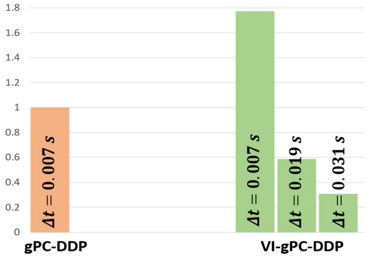


FIG. 10. *Quadrotor: Average elapsed time for the propagation phase of VI-gPC-DPP under different time steps, Δt. The results are normalized with respect to gPC-DDP's corresponding time for $\Delta t = 0.007s$. Utilizing the developed VI allows for coarse discretizations and therefore can reduce computational complexity.*

One major weakness of polynomial chaos theory is that accuracy generally degrades in long-term integration problems. This has to do with the fact that stochastic quantities are represented by finite-dimensional approximations. Adding more coefficients in the gPC expansions may improve accuracy, but it ultimately postpones the issue. In our simulated examples, we focused on a different goal and thus did not consider particularly long horizons. Nevertheless, adaptive multielement methods [33] have been developed in the literature that can deal with the aforementioned issue. These methodologies can additionally handle stochastic discontinuities. One can, therefore, improve our framework by utilizing more sophisticated expansions, as in [33].

Finally, another limitation of gPC theory is that computational complexity increases with the dimensionality of the random inputs. In particular, the required gPC coefficients in (2) grow exponentially with the number of stochastic variables. Although gPC-DDP's scalability is critical, numerical instabilities may occur when dealing with extremely large state spaces. In addition, propagating stochastic dynamics and computing dynamics linearizations become computationally prohibitive is these settings. Nonetheless, the number of significant terms in multidimensional polynomial chaos expansions is usually much smaller than the total number of coefficients K in (2). In this direction, sparse gPC approximation techniques have been designed that can reduce the number of expansion coefficients while retaining accuracy [6, 2]. These can potentially be incorporated into our optimal control methodology and thus allow us to effectively control systems with many uncertain parameters.

8. Conclusion.

In this paper we proposed a new methodology towards stochastic trajectory optimization for systems with uncertainty in the parameters and initial states. Polynomial chaos played a key role since it allowed us to handle a broad class of distributions. The developed optimal control scheme relied on dynamic programming principles and was employed for controlling the statistical behavior of stochastic, high-dimensional, nonlinear systems. Its convergence properties were established by an analysis that extended prior theoretical work. Lastly, it was proven that variational integrators can be formulated when polynomial chaos representations of mechanical systems are considered. We showed that such methods can be naturally used within our algorithm to improve its performance. Simulated examples verified the applicability of the proposed framework and highlighted the benefits of variational integration schemes in stochastic control tasks.

REFERENCES

[1] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 2nd ed., Athena Scientific, 2000.

[2] G. Blatman and B. Sudret, *An adaptive algorithm to build up sparse polynomial chaos expansions for stochastic finite element analysis*, Probab. Engrg. Mech., 25 (2010), pp. 183–197.

[3] G. I. Boutselis, G. De La Torre, and E. A. Theodorou, *Stochastic Optimal Control using polynomial chaos variational integrators*, in 2016 American Control Conference (ACC), Boston, MA, 2016, pp. 6586–6591.

[4] G. I. Boutselis, Y. Pan, G. De La Tore, and E. A. Theodorou, *Stochastic Trajectory Optimization for Mechanical Systems with Parametric Uncertainties*, preprint, https://arxiv.org/abs/1705.05506, 2017.

[5] G. I. Boutselis and E. A. Theodorou, *Spectral variational integrators for trajectory optimization under parametric uncertainties and stochastic disturbances*, in 2016 IEEE 55th Conference on Decision and Control (CDC), Las Vegas, NV, 2016, pp. 2016–2022.

[6] P. G. Constantine, M. S. Eldred, and E. T. Phipps, *Sparse pseudospectral approximation method*, Comput. Methods Appl. Mech. Engrg., 229/232 (2012), pp. 1–12.

[7] M. Deisenroth and C. Rasmussen, *PILCO: A model-based and data-efficient approach to policy search*, in International Conference on Machine Learning, ICML 2011, Omnipress, 2011.

[8] M. Diehl, H. J. Ferreau, and N. Haverbeke, *Efficient Numerical Methods for Nonlinear MPC and Moving Horizon Estimation*, Springer, Berlin, Heidelberg, 2009, pp. 391–417.

[9] P. Dutta and R. Bhattacharya, *Nonlinear estimation with polynomial chaos and higher order moment updates*, in American Control Conference (ACC), IEEE Press, 2010, pp. 3142–3147, https://doi.org/10.1109/ACC.2010.5531023.

[10] M. Eldred and J. Burkardt, *Comparison of non-intrusive polynomial chaos and stochastic collocation methods for uncertainty quantification*, in the 47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition, American Institute of Aeronautics and Astronautics, 2009.

[11] J. Fisher and R. Bhattacharya, *Linear quadratic regulation of systems with stochastic parameter uncertainties*, Automatica J. IFAC, 45 (2009), pp. 2831–2841.

[12] R. Ghanem, *Ingredients for a general purpose stochastic finite elements implementation*, Comput. Methods Appl. Mech. Engrg., 168 (1999), pp. 19–34.

[13] G. H. Golub and J. H. Welsch, *Calculation of Gauss quadrature rules*, Math. Comp., 23 (1969), pp. 221–230.

[14] T. Y. Hou, W. Luo, B. Rozovskii, and H.-M. Zhou, *Wiener chaos expansions and numerical solutions of randomly forced equations of fluid mechanics*, J. Comput. Phys., 216 (2006), pp. 687–706.

[15] F. S. Hover and M. S. Triantafyllou, *Application of polynomial chaos in stability and control*, Automatica J. IFAC, 42 (2006), pp. 789–795.

[16] D. H. Jacobson and D. Q. Mayne, *Differential Dynamic Programming*, Elsevier, 1970.

[17] E. Johnson, J. Schultz, and T. Murphey, *Structured linearization of discrete mechanical systems for analysis and optimal control*, IEEE Trans. Automat. Sci. Engrg., 12 (2015), pp. 140–152.

[18] H. J. Kappen, *Linear theory for control of nonlinear stochastic systems*, Phys. Rev. Lett., 95 (2005), pp. 200–201.

[19] L. Kharevych, W. Yang, Y. Tong, E. Kanso, J. E. Marsden, P. Schröder, and M. Desbrun, *Geometric, variational integrators for computer animation*, in Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2006, pp. 43–51.

[20] M. Kobilarov, M. Desbrun, J. E. Marsden, and G. S. Sukhatme, *A discrete geometric optimal control framework for systems with symmetries*, in Robotics: Science and Systems III, W. Burgard, O. Brock, and C. Stachniss, eds., MIT Press, 2008.

[21] L.-Z. Liao, *Global Convergence of Differential Dynamic Programming and Newton's Method for Discrete-Time Optimal Control*, Technical report, Department of Mathematics, Hong Kong Baptist University, Kowloon Tong, Hong Kong, 1996.

[22] L. Z. Liao and C. A. Shoemaker, *Convergence in unconstrained discrete-time differential dynamic programming*, IEEE Trans. Automat. Control, 36 (1991), pp. 692–706.

[23] N. Mansard, O. Khatib, and A. Kheddar, *A unified approach to integrate unilateral constraints in the stack of tasks*, IEEE Trans. Robotics, 25 (2009), pp. 670–685.

[24] J. E. Marsden and M. West, *Discrete mechanics and variational integrators*, Acta Numer., 10 (2001), pp. 357–514.

[25] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed., Springer, New York, 2006.

[26] Y. Pan and E. Theodorou, *Probabilistic differential dynamic programming*, in Advances in Neural Information Processing Systems (NIPS), 2014, pp. 1907–1915.

[27] J. Pasini and T. Sahai, *Polynomial chaos based uncertainty quantification in Hamiltonian and chaotic systems*, in 52nd Annual IEEE Conference on Decision and Control (CDC), 2013, pp. 1113–1118, https://doi.org/10.1109/CDC.2013.6760031.

[28] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*, MIT Press, 2005.

[29] Y. Tassa, N. Mansard, and E. Todorov, *Control-limited differential dynamic programming*, in IEEE International Conference on Robotics and Automation (ICRA), 2014, pp. 1168–1175, https://doi.org/10.1109/ICRA.2014.6907001.

[30] E. Theodorou, J. Buchli, and S. Schaal, *A generalized path integral control approach to reinforcement learning*, J. Mach. Learn. Res., 11 (2010), pp. 3137–3181.

[31] E. Todorov and W. Li, *A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems*, in Proceedings of the American Control Conference, Vol. 1, IEEE Press, 2005, pp. 300–306, https://doi.org/10.1109/ACC.2005.1469949.

[32] G. D. L. Torre and E. Theodorou, *Stochastic variational integrators for system propagation and linearization*, in IMA Conference on Mathematics of Robotics, Institute of Mathematics and Its Applications, 2015, https://ima.org.uk/proceedings-ima-conference-mathematics-robotics/.

[33] X. Wan and G. E. Karniadakis, *An adaptive multi-element generalized polynomial chaos method for stochastic differential equations*, J. Comput. Phys., 209 (2005), pp. 617–642.

[34] N. Wiener, *The homogeneous chaos*, Amer. J. Math., (1938), pp. 897–936.

[35] D. Xiu, *Numerical Methods for Stochastic Computations: A Spectral Method Approach*, Princeton University Press, 2010.

[36] D. Xiu and G. E. Karniadakis, *The Wiener–Askey polynomial chaos for stochastic differential equations*, SIAM J. Sci. Comput., 24 (2002), pp. 619–644, https://doi.org/10.1137/S1064827501387826.

[37] Y. Xu and P. Vedula, *A moment-based approach for nonlinear stochastic tracking control*, Nonlinear Dynam., 67 (2012), pp. 119–128.

[38] Y. Xu and M. Xin, *Nonlinear stochastic control for space launch vehicles*, IEEE Trans. Aerospace Electronic Systems, 47 (2011), pp. 98–108.