Robotics: Science and Systems 2015
Rome, Italy, July 13-17, 2015

Robust Trajectory Optimization: A Cooperative Stochastic Game Theoretic Approach

Yunpeng Pan
School of Aerospace Engineering
Inst. for Robotics and Intelligent Machines
Georgia Institute of Technology
Atlanta, Georgia, USA
Email: ypan37@gatech.edu

Kaivalya Bakshi
School of Aerospace Engineering
Georgia Institute of Technology
Atlanta, Georgia, USA
Email: kbakshi@gatech.edu

Evangelos A. Theodorou
School of Aerospace Engineering
Inst. for Robotics and Intelligent Machines
Georgia Institute of Technology
Atlanta, Georgia, USA
Email: evangelos.theodorou@ae.gatech.edu

***Abstract*—We present a novel trajectory optimization framework to address the issue of robustness, scalability and efficiency in optimal control and reinforcement learning. Based on prior work in Cooperative Stochastic Differential Game (CSDG) theory, our method performs local trajectory optimization using cooperative controllers. The resulting framework is called Cooperative Game-Differential Dynamic Programming (CG-DDP). Compared to related methods, CG-DDP exhibits improved performance in terms of robustness and efficiency. The proposed framework is also applied in a data-driven fashion for belief space trajectory optimization under learned dynamics. We present experiments showing that CG-DDP can be used for optimal control and reinforcement learning under external disturbances and internal model errors.**

I. Introduction

Model-based trajectory optimization with backward-forward sweeps is a classical and powerful framework in the field of optimal control theory and its applications. It was originally introduced in 1970 under the name "Differential Dynamic Programming (DDP)" [1]. Since then numerous extensions and variations of DDP have been developed in control theory, machine learning and robotics to improve its performance and applicability [2, 3, 4, 5, 6, 7, 8, 9, 10]. The most attractive characteristics of DDP are its computational efficiency and scalability to high-dimensional dynamical systems. However, since it is based on local approximation of the dynamics model and value function, model errors could significantly degrade its performance and therefore restrict its applicability.

An alternative approach to traditional model-based trajectory optimization methods, such us DDP, is Reinforcement Learning (RL) [11]. RL has become one of the major approaches for the development of learning control algorithms with a plethora of applications in robotics and autonomous systems. Compared to the classical trajectory optimization, RL algorithms do not rely on accurate dynamics model, instead they learn optimal policies using data sampled from the physical systems in model-based [12, 13] or model-free [14, 15, 16, 17, 18] fashion. Notable limitations of RL methods include the curse of dimensionality, the need for prior policy parameterization, the requirement for extensive trials performed on real physical systems, and the lack of robustness to disturbances.

Recently, significant efforts have been devoted to solve the aforementioned issues for both RL and trajectory optimization that incorporate explicit representations of uncertainty. In particular, method such as probabilistic model-based RL [13] shows superior efficiency and requires significantly less number of trials for task learning. In addition recent work on trajectory optimization under unknown dynamics [19, 9] has demonstrated impressive learning efficiency and applicability. Most of the aforementioned approaches for RL and trajectory optimization have limited robustness to stochastic disturbances especially in the the cases where these disturbances are far from being Gaussian and zero mean. Disturbances due to system and environment uncertainty are arguably the primary causes of model errors and can degrade the performance of control and learning methodologies when applied to real physical systems.

In control theory the issue of robustness has been addressed in two ways, namely robust control and stochastic optimal control. In modern robust control, polices are designed to bound model uncertainty [20]. While the classical H^∞ control framework has been developed within controls theory, similar ideas have been used in RL as well [21]. The robust control law in [21] is obtained as the solution to a min-max optimization problem. Similarly the minimax-DDP was developed for robust trajectory optimization in [2]. While the minimax-DDP is theoretically appealing due to the explicitly bounded disturbance tolerance, convergence to the desired saddle point is challenging. The family of methods based on min-max criteria also correspond to risk-sensitive optimal control.

In stochastic optimal control, the uncertainty is represented by probability distributions which is usually zero mean and Gaussian. There has been a number of stochastic control methodologies that are based on the exponential transformation of the value function known under the names of Path Integral (PI) control [22] for continuous time, Kullback Leibler (KL) control [23] for discrete time, or more generally linearly solvable optimal control [24]. While these methodologies take into account process noise in the dynamics, their applicability is constrained by the assumption related to control authority and strength of the noise as well as the assumption of Gaussian noise.

In this work, we present an alternative approach to trajectory optimization that is based on the Cooperative Stochastic Differential Game (CSDG) theory. The main characteristics of our approach are summarized as follows: 1) Inspired by the work on differential game formulations and their connection to linear stochastic control problems [25], in this work we propose a CSDG theoretic approach for nonlinear stochastic systems [26]. The resulting algorithm is called Cooperative Game-Differential Dynamic Programming (CG-DDP). 2) CG-DDP finds two optimal control policies under the effects of stochastic disturbances. The two controllers are cooperative and exhibit improved efficiency and robustness to disturbances and stochasticity. 3) CG-DDP can be applied as a model-based RL method such that it learns probabilistic models from sampled data and iteratively optimizes a trajectory in belief spaces. With partial information of a dynamics model, CG-DDP exhibits a superior combination of efficiency and robustness to model errors.

The rest of this paper is organized as follows: In section II, we formulate the control problem as a CSDG for the jump diffusion case. In section III we derive the proposed trajectory optimization framework to solve the CSDG and describe its relation to existing methods. In section IV we introduce a probabilistic model-based RL algorithm based on the proposed CG-DDP framework . In section V we present experimental results and comparative analysis to demonstrate the performance of our method. Finally section VI concludes this paper.

II. Stochastic Cooperative Differential Game Problem Formulation

Consider the following jump diffusion process given by the stochastic differential equation (SDE)

$$\mathrm{d}\mathbf{x} = \mathbf{f}(t,\mathbf{x}(t))\mathrm{d}t + \mathbf{B}(t,\mathbf{x}(t))\boldsymbol{\tau}(t)\mathrm{d}t + \mathbf{C}(t)\mathrm{d}\boldsymbol{\omega} + \mathbf{H}(t)\mathrm{d}\mathbf{p}, \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^{n_x}$ is the state, $\boldsymbol{\tau} \in \mathbb{R}^{n_u}$ is the control, $\boldsymbol{\omega} \in \mathbb{R}^{n_w}$ and $\mathbf{p} \in \mathbb{R}^{n_p}$ are standard Wiener and Poisson processes. In addition the dimensionality of the terms $\mathbf{f}, \mathbf{B}, \mathbf{C}, \mathbf{H}$ in (1) is defined as $\mathbf{f} \in \mathbb{R}^{n_x}$, $\mathbf{B} \in \mathbb{R}^{n_x \times n_u}$, $\mathbf{C} \in \mathbb{R}^{n_x \times n_w}$ and $\mathbf{H} \in \mathbb{R}^{n_x \times n_p}$. Moreover let $\mathbf{h}^j$ be the j^{th} the column vector in $\mathbf{H}$. For the Poisson process $\mathbb{E}[\mathrm{d}\mathbf{p}^j] = \lambda^j \mathrm{d}t$ and $\lambda^j \in \mathbb{R}\ \forall 1 \leq j \leq n_p$. We denote $\boldsymbol{\lambda} = [\lambda]_{n_p \times 1}$ the rate with which jumps occur. Consider a finite-horizon stochastic optimal control problem for the cost functional

$$J(t,\mathbf{x};\pi) = \mathbb{E}\Big[\underbrace{q(\mathbf{x}(T))}_{\textbf{Terminal cost}} + \int_t^T \underbrace{\mathcal{L}\big(t,\mathbf{x}(t),\pi(t,\mathbf{x}(t))\big)}_{\textbf{Running cost}}\mathrm{d}t\Big], \tag{2}$$

where the goal is to find a control policy $\boldsymbol{\tau} = \pi(t,\mathbf{x}(t))$ that minimizes the total cost accumulated over the time horizon T. Here we define two control variables $\mathbf{u} \in \mathbb{R}^{n_m}$ and $\mathbf{v} \in \mathbb{R}^{n_m}$ such that the original control input has been split into two parts: $\boldsymbol{\tau} = \mathbf{u} + \mathbf{v}$. Next we show that the original control problem can be reformulated as a two-player CSDG with a given time horizon T and players $\mathbf{u}$,$\mathbf{v}$. A simple illustration is shown in Fig.(1). Functions $\pi^{\mathbf{u}}$, $\pi^{\mathbf{v}}$ denote strategies (policies) for $\mathbf{u}$ and $\mathbf{v}$ from t to T. The notation $\pi^{\mathbf{u}} = \emptyset$ implies $\mathbf{u}(t), ..., \mathbf{u}(T) = 0$ and $\pi^{\mathbf{v}} = \emptyset$ is defined similarly. In addition the notation $\pi^u \cup \pi^v$ denotes a cooperative game theoretic control strategy. Next we define the payoff function for each individual player as

$$\mathcal{V}^{\mathbf{u}}(\pi^{\mathbf{u}} \cup \pi^{\mathbf{v}}) = J^{\mathbf{u}}(t,\mathbf{x};\emptyset) - J^{\mathbf{u}}(t,\mathbf{x};\pi^{\mathbf{u}},\pi^{\mathbf{v}}),\ \forall \pi^{\mathbf{u}},\pi^{\mathbf{v}},$$

where

$$J^{\mathbf{u}}(t,\mathbf{x};\pi^{\mathbf{u}},\pi^{\mathbf{v}}) = \mathbb{E}\Big[q^{\mathbf{u}}(\mathbf{x}(T)) + \int_t^T \mathcal{L}^{\mathbf{u}}(t,\mathbf{x}(t),\pi^{\mathbf{u}},\pi^{\mathbf{v}})\mathrm{d}t\Big],$$

and $J^{\mathbf{u}}(t,\mathbf{x};\emptyset)$ corresponds to the cost with both controls taken to be zero. Payoff $\mathcal{V}^{\mathbf{v}}(\pi^{\mathbf{u}} \cup \pi^{\mathbf{v}})$ can be defined similarly. The coalition or cooperative payoff function is defined as

$$\mathcal{V}(\pi^{\mathbf{u}} \cup \pi^{\mathbf{v}}) = J(t,\mathbf{x};\emptyset) - J(t,\mathbf{x};\pi^{\mathbf{u}} \cup \pi^{\mathbf{v}}),$$

where

$$J(t,\mathbf{x};\pi^{\mathbf{u}} \cup \pi^{\mathbf{v}}) = \mathbb{E}\Big[q(\mathbf{x}(T)) + \int_t^T \mathcal{L}(t,\mathbf{x}(t),\pi^{\mathbf{u}},\pi^{\mathbf{v}})\mathrm{d}t\Big],$$

and $J(t,\mathbf{x};\emptyset)$ is the uncontrolled cost. Therefore the coalition $\pi^{\mathbf{u}} \cup \pi^{\mathbf{v}}$ has a real valued payoff assigned by the cooperative payoff function for which $\mathcal{V}(\emptyset) = 0$. Both $\mathbf{u}$ and $\mathbf{v}$ try to maximize the assigned individual payoff function at t. Based on the definition of a coalition game in [27] the above stated problem defines a coalition game with players $\mathbf{u}$ and $\mathbf{v}$. We follow [26] and denote this two player *C*ooperative *G*ame by $CG(\mathbf{x}, T-t)$ played at current state $\mathbf{x}$ for the time interval $[t,T]$, where both players $\mathbf{u}$ and $\mathbf{v}$ agree to cooperate based on an optimality principle. The optimality principle for a cooperative scheme includes: i) an agreement on a set of cooperative strategies/controls, and ii) a mechanism to distribute total payoff among players, iii) group rationality requires the players to seek a set of optimal cooperative strategies $\pi^{\mathbf{u}}$ and $\pi^{\mathbf{v}}$ that justify the participation of each in the $CG(\mathbf{x}, T-t)$. This is mathematically expressed as

$$\mathcal{V}(\pi^{\mathbf{u}*} \cup \pi^{\mathbf{v}*}) \geq \mathcal{V}(\emptyset \cup \pi^{\mathbf{v}}), \mathcal{V}(\pi^{\mathbf{u}} \cup \emptyset),\ \forall \pi^{\mathbf{u}},\pi^{\mathbf{v}}.$$

Considering the fact that $J(t,\mathbf{x};\mathbf{0})$ does not depend on the selected strategies, the solution to $CG(\mathbf{x}, T-t)$ is therefore to solve

$$\underset{\pi^{\mathbf{u}},\pi^{\mathbf{v}}}{\operatorname{argmin}}\ \mathbb{E}\Big[\sum_{j=\{\mathbf{u},\mathbf{v}\}} q^j(\mathbf{x}(T)) + \int_t^T \sum_{j=\{\mathbf{u},\mathbf{v}\}} \mathcal{L}^j(t,\mathbf{x}(t),\pi^{\mathbf{u}},\pi^{\mathbf{v}})\mathrm{d}t\Big]. \tag{3}$$

We choose in this case to distribute the individual payoff functions s.t.

$$q^{\mathbf{u}}(\mathbf{x}) + q^{\mathbf{v}}(\mathbf{x}) = q(\mathbf{x}), \quad \mathcal{L}^{\mathbf{u}}(\mathbf{x}) + \mathcal{L}^{\mathbf{v}}(\mathbf{x}) = \mathcal{L}(\mathbf{x}),$$

so that the solution is to solve

$$\underset{\pi^{\mathbf{u}},\pi^{\mathbf{v}}}{\operatorname{argmin}}\ \mathbb{E}\Big[q(\mathbf{x}(T)) + \int_t^T \mathcal{L}(t,\mathbf{x}(t),\pi^{\mathbf{u}},\pi^{\mathbf{v}})\mathrm{d}t\Big], \tag{4}$$

subject to (1). This problem is called the *C*ooperative *S*tochastic *D*ifferential *G*ame in Game Theory literature [28,

26], and is denoted by *CSDG*$(\mathbf{x}, T-t)$ the solution to which is stated in the following theorem.

Theorem 2.1: A set of controls $[\mathbf{u}^*(t), \mathbf{v}^*(t)]$ *provides an optimal solution to the problem CSDG*$(\mathbf{x}, T-t)$ *if there exists a continuously differentiable function* $V(t,\mathbf{x}) : [t,T] \times \mathbb{R}^n_x \to \mathbb{R}$ *which satisfies the PDE*

$$-\frac{\partial V}{\partial t}(t,\mathbf{x}) - \frac{1}{2}\mathrm{tr}(\mathbf{C}\mathbf{C}^{\mathrm{T}} V_{\mathbf{xx}}(t,\mathbf{x})) - \sum_{j=1}^{n_p}(V(t,\mathbf{x}+\mathbf{h}^j(t)) - V(t,\mathbf{x}))\lambda^j(t) = \min_{\mathbf{u}(t),\mathbf{v}(t)}\Big(\mathcal{L}(t,\mathbf{x},\mathbf{u},\mathbf{v}) + V_{\mathbf{x}}^T(\mathbf{f}(\mathbf{x}) + \mathbf{B}(\mathbf{x})(\mathbf{u}+\mathbf{v}))\Big) \quad (5)$$

with the boundary condition $V(T,\mathbf{x}) = q(T,\mathbf{x})$.

Theorem 2.1 is an extension of [26] for the case of minimum game for jump diffusion processes. The related proof is included in the supplementary material of this work. This theorem provides necessary conditions for the existence of cooperative control strategies that are group rational. The function satisfying the above PDE is the well-known value function defined as

$$V(t,\mathbf{x}) = \min_{\mathbf{u},\mathbf{v}} \mathbb{E}\big[q(\mathbf{x}(T)) + \int_t^T \mathcal{L}(t,\mathbf{x}(t),\mathbf{u},\mathbf{v})\mathrm{d}t \big| \mathbf{x}(t) = \mathbf{x}\big].$$

Given the existence of such a value function one has to solve this PDE in order to obtain the optimal control strategy at a given $(t,\mathbf{x})$. However, finding the solution for the PDE in (5) is computationally intractable especially for systems with many degree of freedom such as robotic systems. This is the so-called the *curse of dimensionality*. One way to bypass this is to rely on approximation methods that solve this problem locally, in particular by approximating the value function along nominal trajectories. DDP is known to be one of the most efficient approaches to value function approximation while it features second order convergence [29] and scales to high dimensional robotic systems [7]. In this work we propose a scheme to solve the CSDG locally by transforming the problem to discrete time and working with second order approximations of value/cost functions, and first order approximation of stochastic dynamics. The resulting framework is called Cooperative Game-Differential Dynamic Programming (CG-DDP). For the rest of the analysis we use the discretized representation of the system in (1)

$$\mathbf{x}_{t+\mathrm{d}t} = \mathbf{x}_t + \mathbf{f}(\mathbf{x}_t)\mathrm{d}t + \mathbf{B}(\mathbf{x}_t)(\mathbf{u}_t+\mathbf{v}_t)\mathrm{d}t + \mathbf{C}(t)\mathrm{d}\boldsymbol{\omega} + \mathbf{H}(t)\mathrm{d}\mathbf{p}. \quad (6)$$

To simplify notation we use subscripts of variables to denote discrete time step. Next we show how to derive the optimal policies for both $\mathbf{u}_t$ and $\mathbf{v}_t$ to solve (5).

III. The Cooperative Game based Trajectory Optimization Framework

A. Backward-Sweep: Optimal Cost-to-Go

The proposed trajectory optimization method is rooted in the Dynamic Programming principle. First we consider a variation

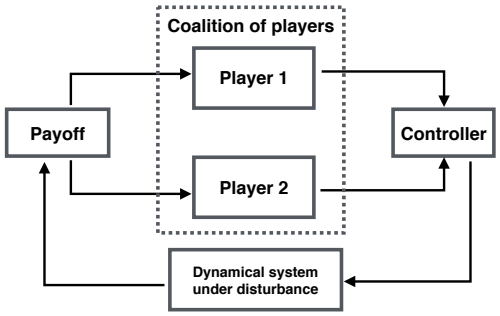

Fig. 1: Cooperative stochastic game theoretic controller. The controller consists of a coalition of two players.

of the Bellman equation with both $\mathbf{u}_t$ and $\mathbf{v}_t$

$$V(t,\mathbf{x}_t) = \min_{\mathbf{u}_t,\mathbf{v}_t} \underbrace{\mathbb{E}\Big[\mathcal{L}(t,\mathbf{x}_t,\mathbf{u}_t,\mathbf{v}_t) + V(t+\mathrm{d}t,\mathbf{x}_{t+\mathrm{d}t})\Big]}_{Q(\mathbf{x}_t,\mathbf{u}_t,\mathbf{v}_t)}. \quad (7)$$

The goal is to find feedback policies such that the two-player pair $(\mathbf{u}_t,\mathbf{v}_t)$ minimize the total cost. First we build a local model in the neighborhood of a nominal trajectory. For this purpose we create a local model of the dynamics around $\bar{\mathbf{x}}_t, \bar{\mathbf{u}}_t, \bar{\mathbf{v}}_t$.

$$\delta\mathbf{x}_{t+\mathrm{d}t} = \underbrace{\mathbf{A}_t\delta\mathbf{x}_t + \mathbf{B}_t\big(\delta\mathbf{u}_t + \delta\mathbf{v}_t\big) + \mathbf{C}_t\mathrm{d}\boldsymbol{\omega}}_{\delta\mathbf{x}^{\mathrm{cont}}_{t+\mathrm{d}t}} + \underbrace{\mathbf{H}_t\mathrm{d}\mathbf{p}}_{\delta\mathbf{x}^{\mathrm{jump}}_{t+\mathrm{d}t}}, \quad (8)$$

where $\delta\mathbf{x}_t = \mathbf{x}_t - \bar{\mathbf{x}}_t, \delta\mathbf{u}_t = \mathbf{u}_t - \bar{\mathbf{u}}_t$ and $\delta\mathbf{v}_t = \mathbf{v}_t - \bar{\mathbf{v}}_t$ are defined as the deviations from the nominal trajectory, $\mathbf{A}_t = (I_{n_x\times n_x} + \mathbf{f}_{\mathbf{x}}(\bar{\mathbf{x}_t}))\mathrm{d}t$, $\mathbf{B}_t = \mathbf{B}(t,\bar{\mathbf{x}_t})\mathrm{d}t$, $\mathbf{C}_t = \mathbf{C}(t)\sqrt{\mathrm{d}t}$ and $\mathbf{H}_t = \mathbf{H}(t)\mathrm{d}t$. To evaluate the expectation under minimization in (7) we use the Ito stochastic chain rule for jump diffusion process [30, 31] and we get with $\mathrm{d}t$ precision

$$\begin{aligned}
&\mathbb{E}\Big[V(\bar{\mathbf{x}}_{t+\mathrm{d}t} + \delta\mathbf{x}_{t+\mathrm{d}t})\Big] \overset{Ito}{=} \mathbb{E}\Big[V(\bar{\mathbf{x}}_{t+\mathrm{d}t}) \\
&+ V_{\mathbf{x}}^{\mathrm{T}}(\bar{\mathbf{x}}_{t+\mathrm{d}t})\delta\mathbf{x}^{cont}_{t+\mathrm{d}t} + \frac{1}{2}\delta\mathbf{x}^{cont}_{t+\mathrm{d}t}{}^{\mathrm{T}} V_{\mathbf{xx}}\delta\mathbf{x}^{cont}_{t+\mathrm{d}t} \\
&+ \sum_{j=1}^{n_p}\Big(V(\bar{\mathbf{x}}_{t+\mathrm{d}t} + \mathbf{h}_t^{i,j}) - V(\bar{\mathbf{x}}_{t+\mathrm{d}t})\Big)\mathrm{d}\mathbf{p}_t^j | \mathbf{x}_t\Big] \\
&= V(\bar{\mathbf{x}}_{t+\mathrm{d}t}) + V_{\mathbf{x}}^{\mathrm{T}}(\bar{\mathbf{x}}_{t+\mathrm{d}t})\Big(\mathbf{A}_t\delta\mathbf{x}_t + \mathbf{B}_t(\delta\mathbf{u}_t + \delta\mathbf{v}_t)\Big) \\
&+ \frac{1}{2}\Big(\delta\mathbf{x}_t^{\mathrm{T}}\mathbf{A}_t^{\mathrm{T}}V_{\mathbf{xx}}\mathbf{A}_t\delta\mathbf{x}_t + (\delta\mathbf{u}_t+\delta\mathbf{v}_t)^{\mathrm{T}}\mathbf{B}_t^{\mathrm{T}}V_{\mathbf{xx}}\mathbf{B}_t(\delta\mathbf{u}_t+\delta\mathbf{v}_t) \\
&+ (\delta\mathbf{u}_t+\delta\mathbf{v}_t)^{\mathrm{T}}\mathbf{B}_t^{\mathrm{T}}V_{\mathbf{xx}}\mathbf{A}_t\delta\mathbf{x}_t + \delta\mathbf{x}_t^{\mathrm{T}}\mathbf{A}_t^{\mathrm{T}}V_{\mathbf{xx}}\mathbf{B}_t(\delta\mathbf{u}_t+\delta\mathbf{v}_t)\Big) \\
&+ \frac{1}{2}\mathrm{tr}\Big(V_{\mathbf{xx}}\mathbf{C}_t\mathbf{C}_t^{\mathrm{T}}\Big) + \sum_{j=1}^{n_p}\Big(V_{\mathbf{x}}^{\mathrm{T}}\mathbf{h}_t^j\lambda_t^j + (\mathbf{h}_t^j)^{\mathrm{T}}V_{\mathbf{xx}}\mathbf{h}_t^j\lambda_t^j\Big)\mathrm{d}t
\end{aligned} \quad (9)$$

where the nonlinear jump term of the value function has been approximated till its second order Taylor series expansion. The detailed derivation of the terms appearing due to the continuous part of the deviation in the nominal trajectory can be found in [32].

Next we build a local quadratic model of the value function by expanding the Q-function up to the second order

$$Q(\bar{\mathbf{x}}_t+\delta\mathbf{x}_t,\bar{\mathbf{u}}_t+\delta\mathbf{u}_t,\bar{\mathbf{v}}_t+\delta\mathbf{v}_t)\approx Q_0+Q_{\mathbf{x}}\delta\mathbf{x}_t+Q_{\mathbf{u}}\delta\mathbf{u}_t + Q_{\mathbf{v}}\delta\mathbf{v}_t+\frac{1}{2}\begin{bmatrix}\delta\mathbf{x}_t\\ \delta\mathbf{u}_t\\ \delta\mathbf{v}_t\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}Q_{\mathbf{xx}} & Q_{\mathbf{xu}} & Q_{\mathbf{xv}}\\ Q_{\mathbf{ux}} & Q_{\mathbf{uu}} & Q_{\mathbf{uv}}\\ Q_{\mathbf{vx}} & Q_{\mathbf{vu}} & Q_{\mathbf{vv}}\end{bmatrix}\begin{bmatrix}\delta\mathbf{x}_t\\ \delta\mathbf{u}_t\\ \delta\mathbf{v}_t\end{bmatrix}. \tag{10}$$

where all the Q-related terms are given as

$$\begin{aligned}
Q_0 &= V_{t+\mathrm{d}t}+\frac{1}{2}\mathrm{tr}(V_{\mathbf{xx}}\mathbf{C}_t\mathbf{C}_t^{\mathrm{T}}) + \sum_{j=1}^{n_p}\Big(V_{\mathbf{x}}^{\mathrm{T}}\mathbf{h}_t^j\lambda_t^j+(\mathbf{h}_t^j)^{\mathrm{T}}V_{\mathbf{xx}}\mathbf{h}_t^j\lambda_t^j\Big)\mathrm{d}t,\\
Q_{\mathbf{x}} &= V_{\mathbf{x}}^{\mathrm{T}}\mathbf{A}_t+\mathcal{L}_{\mathbf{x}},\ Q_{\mathbf{u}}=V_{\mathbf{x}}^{\mathrm{T}}\mathbf{B}_t+\mathcal{L}_{\mathbf{u}},\ Q_{\mathbf{v}}=V_{\mathbf{x}}^{\mathrm{T}}\mathbf{B}_t+\mathcal{L}_{\mathbf{v}},\\
Q_{\mathbf{xx}} &= \mathbf{A}_t^{\mathrm{T}}V_{\mathbf{xx}}\mathbf{A}_t+\mathcal{L}_{\mathbf{xx}},\ Q_{\mathbf{xu}}=\mathbf{A}^{\mathrm{T}}V_{\mathbf{xx}}\mathbf{B}_t+\mathcal{L}_{\mathbf{xu}},\\
Q_{\mathbf{xv}} &= \mathbf{A}_t^{\mathrm{T}}V_{\mathbf{xx}}\mathbf{B}_t+\mathcal{L}_{\mathbf{xv}},\ Q_{\mathbf{uu}}=\mathbf{B}_t^{\mathrm{T}}V_{\mathbf{xx}}\mathbf{B}_t+\mathcal{L}_{\mathbf{uu}},\\
Q_{\mathbf{vv}} &= \mathbf{B}_t^{\mathrm{T}}V_{\mathbf{xx}}\mathbf{B}_t+\mathcal{L}_{\mathbf{vv}},\ Q_{\mathbf{uv}}=\mathbf{B}_t^{\mathrm{T}}V_{\mathbf{xx}}\mathbf{B}_t+\mathcal{L}_{\mathbf{uv}},
\end{aligned}$$

where $V_t=V(t,\bar{\mathbf{x}}_t)$. In order to find the optimal policies for $\delta\mathbf{u}_t$ and $\delta\mathbf{v}_t$ such that the second-order expansion of the Q-function is minimized, we take the gradients of (10) with respect to $\delta\mathbf{u}_t$ and $\delta\mathbf{v}_t$

$$\begin{aligned}
&Q_{\delta\mathbf{u}}\big(\mathbf{x}_t+\delta\mathbf{x}_t,\mathbf{u}_t+\delta\mathbf{u}_t,\mathbf{v}_t+\delta\mathbf{v}_t\big)=0\\
&\Rightarrow \delta\mathbf{u}_t^*=-Q_{\mathbf{uu}}^{-1}\big(Q_{\mathbf{ux}}\delta\mathbf{x}_t+Q_{\mathbf{uv}}\delta\mathbf{v}_t+Q_{\mathbf{u}}\big),\\
&Q_{\delta\mathbf{v}}\big(\mathbf{x}_t+\delta\mathbf{x}_t,\mathbf{u}_t+\delta\mathbf{u}_t,\mathbf{v}_t+\delta\mathbf{v}_t\big)=0\\
&\Rightarrow \delta\mathbf{v}_t^*=-Q_{\mathbf{vv}}^{-1}\big(Q_{\mathbf{vx}}\delta\mathbf{x}_t+Q_{\mathbf{vu}}\delta\mathbf{u}_t+Q_{\mathbf{v}}\big).
\end{aligned} \tag{11}$$

Solving the system of equations in (11) results in the expressions

$$\begin{aligned}
\delta\mathbf{u}_t^* &= \underbrace{-\Big(Q_{\mathbf{uu}}-Q_{\mathbf{uv}}Q_{\mathbf{vv}}^{-1}Q_{\mathbf{vu}}\Big)^{-1}\Big(Q_{\mathbf{u}}-Q_{\mathbf{uv}}Q_{\mathbf{vv}}^{-1}Q_{\mathbf{v}}\Big)}_{\mathbf{I_u}}\\
&\underbrace{-\Big(Q_{\mathbf{uu}}-Q_{\mathbf{uv}}Q_{\mathbf{vv}}^{-1}Q_{\mathbf{vu}}\Big)^{-1}\Big(Q_{\mathbf{ux}}-Q_{\mathbf{uv}}Q_{\mathbf{vv}}^{-1}Q_{\mathbf{vx}}\Big)}_{\mathbf{L_u}}\delta\mathbf{x}_t\\
\delta\mathbf{v}_t^* &= \underbrace{-\Big(Q_{\mathbf{vv}}-Q_{\mathbf{vu}}Q_{\mathbf{uu}}^{-1}Q_{\mathbf{uv}}\Big)^{-1}\Big(Q_{\mathbf{v}}-Q_{\mathbf{vu}}Q_{\mathbf{uu}}^{-1}Q_{\mathbf{u}}\Big)}_{\mathbf{I_v}}\\
&\underbrace{-\Big(Q_{\mathbf{vv}}-Q_{\mathbf{vu}}Q_{\mathbf{uu}}^{-1}Q_{\mathbf{uv}}\Big)^{-1}\Big(Q_{\mathbf{vx}}-Q_{\mathbf{vu}}Q_{\mathbf{uu}}^{-1}Q_{\mathbf{ux}}\Big)}_{\mathbf{L_v}}\delta\mathbf{x}_t.
\end{aligned} \tag{12}$$

where the terms $\mathbf{I_u},\mathbf{L_u}$ and $\mathbf{I_v},\mathbf{L_v}$ are the feedforward and feedback terms of the optimal control updates $\delta\mathbf{u}_t^*$ and $\delta\mathbf{v}_t^*$. By plugging the optimal control updates $\delta\mathbf{u}_t^*$ and $\delta\mathbf{v}_t^*$ into the value function, we can split the value function into zero, first and second order terms in $\delta\mathbf{x}_t$

$$V(\bar{\mathbf{x}}_t+\delta\mathbf{x}_t)=V_t+V_{\mathbf{x}}^{\mathrm{T}}\delta\mathbf{x}_t+\frac{1}{2}\delta\mathbf{x}_t^{\mathrm{T}}V_{\mathbf{xx}}\delta\mathbf{x}_t, \tag{13}$$

where V_t, $V_{\mathbf{x}}$ and $V_{\mathbf{xx}}$ are computed as

$$\begin{aligned}
V_t =& Q_0+\mathbf{l}_{\mathbf{u}}^{\mathrm{T}}Q_{\mathbf{u}}+\mathbf{l}_{\mathbf{v}}^{\mathrm{T}}Q_{\mathbf{v}}\\
&+\frac{1}{2}\Big(\mathbf{l}_{\mathbf{u}}^{\mathrm{T}}Q_{\mathbf{uu}}\mathbf{l}_{\mathbf{u}}+\mathbf{l}_{\mathbf{v}}^{\mathrm{T}}Q_{\mathbf{vv}}\mathbf{l}_{\mathbf{v}}+\mathbf{l}_{\mathbf{u}}^{\mathrm{T}}Q_{\mathbf{uu}}\mathbf{l}_{\mathbf{v}}+\mathbf{l}_{\mathbf{v}}^{\mathrm{T}}Q_{\mathbf{vu}}\mathbf{l}_{\mathbf{u}}\Big),\\
V_{\mathbf{x}} =& Q_{\mathbf{x}}+\mathbf{L}_{\mathbf{u}}^{\mathrm{T}}Q_{\mathbf{u}}+\mathbf{L}_{\mathbf{v}}^{\mathrm{T}}Q_{\mathbf{v}}+Q_{\mathbf{xu}}\mathbf{l}_{\mathbf{u}}+Q_{\mathbf{xv}}\mathbf{l}_{\mathbf{v}}\\
&+\mathbf{L}_{\mathbf{u}}^{\mathrm{T}}Q_{\mathbf{uu}}\mathbf{l}_{\mathbf{u}}+\mathbf{L}_{\mathbf{v}}^{\mathrm{T}}Q_{\mathbf{vv}}\mathbf{l}_{\mathbf{v}}+\mathbf{L}_{\mathbf{u}}^{\mathrm{T}}Q_{\mathbf{uv}}\mathbf{l}_{\mathbf{v}}+\mathbf{L}_{\mathbf{v}}^{\mathrm{T}}Q_{\mathbf{vu}}\mathbf{l}_{\mathbf{u}},\\
V_{\mathbf{xx}} =& Q_{\mathbf{xx}}+\mathbf{L}_{\mathbf{ux}}^{\mathrm{T}}Q_{\mathbf{ux}}+\mathbf{L}_{\mathbf{v}}^{\mathrm{T}}Q_{\mathbf{v}}+Q_{\mathbf{xu}}\mathbf{l}_{\mathbf{u}}+Q_{\mathbf{xv}}\mathbf{l}_{\mathbf{v}}\\
&+\mathbf{L}_{\mathbf{u}}^{\mathrm{T}}Q_{\mathbf{uu}}\mathbf{L}_{\mathbf{u}}+\mathbf{L}_{\mathbf{v}}^{\mathrm{T}}Q_{\mathbf{vv}}\mathbf{L}_{\mathbf{v}}+\mathbf{L}_{\mathbf{u}}^{\mathrm{T}}Q_{\mathbf{uv}}\mathbf{L}_{\mathbf{v}}+\mathbf{L}_{\mathbf{v}}^{\mathrm{T}}Q_{\mathbf{vu}}\mathbf{L}_{\mathbf{u}}.
\end{aligned} \tag{14}$$

Since the optimal cost-to-go is computed backward in time, this computational scheme is called the backward-sweep in trajectory optimization. Next we introduce the forward-sweep scheme of CG-DDP.

B. Update control laws and forward sweep

In eq.(12) we have found the optimal feedback laws for correction terms $\delta\mathbf{u}_t$ and $\delta\mathbf{v}_t$. We compute the optimal controllers for the next iteration as

$$\begin{aligned}
\mathbf{u}_t^* &= \mathbf{u}_t+\delta\mathbf{u}_t^*=\mathbf{u}_t+\mathbf{I}_{\mathbf{u}}+\mathbf{L}_{\mathbf{u}}\delta\mathbf{x}_t,\\
\mathbf{v}_t^* &= \mathbf{v}_t+\delta\mathbf{v}_t^*=\mathbf{v}_t+\mathbf{I}_{\mathbf{v}}+\mathbf{L}_{\mathbf{v}}\delta\mathbf{x}_t.
\end{aligned} \tag{15}$$

Obviously these are linear, time-varying policies but without any a-priori policy parameterization. The results in eq.(15) are locally optimal controls for the original nonlinear system in the vicinity of the nominal trajectory $\bar{\mathbf{x}}$. To obtain an optimal trajectory we iteratively update the nominal trajectory by applying the optimized policies. At each iteration the optimized trajectory becomes the new nominal for the next iteration. Similar to standard DDP, the proposed optimization approach is a second-order method that relies on the Hessian matrices that appear in eq.(12) and are expressed as

$$\begin{aligned}
\mathbf{H}_{\mathbf{u}} &= Q_{\mathbf{uu}}-Q_{\mathbf{uv}}Q_{\mathbf{vv}}^{-1}Q_{\mathbf{vu}},\\
\mathbf{H}_{\mathbf{v}} &= Q_{\mathbf{vv}}-Q_{\mathbf{vu}}Q_{\mathbf{uu}}^{-1}Q_{\mathbf{uv}}.
\end{aligned} \tag{16}$$

The cost-to-go decreases in the direction of $\delta\mathbf{u}_t$ and $\delta\mathbf{v}_t$ for positive definite $\mathbf{H}_{\mathbf{u}}$ and $\mathbf{H}_{\mathbf{v}}$.

One of the attractive characteristics of the proposed framework is the robustness against disturbances. Besides the CSDG theory, this characteristic can be also interpreted from the structure $\mathbf{H}_{\mathbf{u}}$ and $\mathbf{H}_{\mathbf{v}}$ in (16). In particular, we consider the case where $Q_{\mathbf{uu}}$ and $Q_{\mathbf{vv}}$ are positive (semi)definite along the nominal trajectory $\bar{\mathbf{u}}$ and $\bar{\mathbf{v}}$. In the classical DDP, the term $\mathbf{H}_{\mathbf{u}}=Q_{\mathbf{uu}}$. In our case, given the positive definiteness of $Q_{\mathbf{uu}},Q_{\mathbf{vv}},Q_{\mathbf{uu}}^{-1},Q_{\mathbf{vv}}^{-1}$, it is easy to show that $\mathbf{H}_{\mathbf{u}}$ will have relatively small eigenvalues and therefore large $\mathbf{H}_{\mathbf{u}}^{-1}$ leads to high feedback gain. The same reasoning applies to the case of $\mathbf{H}_{\mathbf{v}}$. In order to improve numerical stability we would like to ensure positive definiteness of both $\mathbf{H}_{\mathbf{u}}$ and $\mathbf{H}_{\mathbf{v}}$. In this work we employ the Levenberg-Marquardt trick (also used in

[3]). When $\mathbf{H_u}$ has negative eigenvalues, we first compute the eigenvalue decomposition, i.e., $[\mathbf{V}, \mathbf{D}]$ =eig($\mathbf{H_u}$) and replace all negative elements of $\mathbf{D}$ with 0. Then a small positive value λ is added to $\mathbf{D}$. The modified matrix is obtained as $\tilde{\mathbf{H}}_\mathbf{u} = \mathbf{V}\mathbf{D}\mathbf{V}^\mathrm{T}$. Its inverse is computed as $\tilde{\mathbf{H}}_\mathbf{u}^{-1} = \mathbf{V}\mathbf{D}^{-1}\mathbf{V}^\mathrm{T}$. Matrix $\mathbf{H_v}$ can be modified similarly.

In the proposed framework we implement line search by adding a parameter $\varepsilon > 0$ such that $\delta\mathbf{u}_t^* = \varepsilon\mathbf{I_u} + \mathbf{L_u}\delta\mathbf{x}_t$ and $\delta\mathbf{v}_t^* = \varepsilon\mathbf{I_v} + \mathbf{L_v}\delta\mathbf{x}_t$. Initially $\varepsilon = 1$, when the trajectory generated by the learned policy has a higher cost than the current one, the policy would be rejected and decrease ε. Whenever the policy is accepted we reset $\varepsilon = 1$. This trick has also been used in [3, 6] to encourage convergence.

The optimization is performed iteratively only in internal simulation. When interacting with a real physical system we have to control a strongly stochastic dynamics (6). The control policy implemented in the physical system (6) is a combination of the optimized policies

$$\boldsymbol{\tau}_t^* = \pi(\mathbf{x}_t) = \underbrace{\bar{\mathbf{u}}_t^* + \mathbf{L_u}\big(\mathbf{x}_t - \bar{\mathbf{x}}_t^*\big)}_{\mathbf{u}_t^*} + \underbrace{\bar{\mathbf{v}}_t^* + \mathbf{L_v}\big(\mathbf{x}_t - \bar{\mathbf{x}}_t^*\big)}_{\mathbf{v}_t^*}. \tag{17}$$

Where $\bar{\mathbf{x}}_t^*, \bar{\mathbf{u}}_t^*, \bar{\mathbf{v}}_t^*$ are the optimized trajectory and controllers obtained from simulation (the final nominal trajectory). $\mathbf{x}_t$ is the actual state. We do not apply the open-loop policy $\mathbf{I_u}$ and $\mathbf{I_v}$ whose magnitudes usually vanish (or become very small) during the final stage of optimization.

C. Summary of Algorithm

The proposed optimization scheme (Algorithm 1) can be summarized as follows: 1) Initially, apply random controls to the dynamical system to obtain the nominal trajectory $\bar{\mathbf{x}}, \bar{\mathbf{u}}, \bar{\mathbf{v}}$. In particular $\bar{\mathbf{v}}$ is initialized with an estimate of the worst case disturbances $\mathbf{w}$. 2) Linearize the belief dynamics around $\bar{\mathbf{x}}, \bar{\mathbf{u}}, \bar{\mathbf{v}}$ (Sec.IV-B). 3) Backward-sweep to obtain the policy parameters $\mathbf{I_u}, \mathbf{I_v}, \mathbf{L_u}, \mathbf{L_v}$. We use the Levenberg-Marquardt method to ensure positive definiteness of $\mathbf{H_u}$ and $\mathbf{H_v}$ (Sec.III-A). 4) Forward-sweep to obtain the a new trajectory $\bar{\mathbf{u}}, \bar{\mathbf{v}}, \bar{\mathbf{x}}$ which are set to be the new nominal trajectory. An estimate of stochastic disturbances $\mathbf{w}$ may be added to $\bar{\mathbf{v}}$ (or $\bar{\mathbf{u}}$) during optimization. We use line search to encourage convergence (Sec.III-B). 5) The iterative process is applied until convergence. And the control policy (eq.17) is applied to the system to generate the optimized trajectory.

Algorithm 1 CG-DDP

1: **Initialization:** Choose $\bar{\mathbf{u}}, \bar{\mathbf{v}}, \bar{\mathbf{x}}$ randomly.
2: **repeat**
3: Linearize the dynamics model around $\bar{\mathbf{x}}, \bar{\mathbf{u}}, \bar{\mathbf{v}}$.
4: Backward-sweep: compute $\mathbf{I_u}, \mathbf{I_v}, \mathbf{L_u}, \mathbf{L_v}$ (III-A).
5: Forward-sweep: compute $\mathbf{u}^*$,$\mathbf{v}^*$,$\mathbf{x}^*$ (III-B).
6: Update nominal trajectory: $(\bar{\mathbf{u}}, \bar{\mathbf{v}}, \bar{\mathbf{x}})$ =$(\mathbf{u}^*, \mathbf{v}^*, \mathbf{x}^*)$
7: **until** Convergence.
8: **return** $\bar{\mathbf{u}}, \bar{\mathbf{v}}, \bar{\mathbf{x}}, \mathbf{I_u}, \mathbf{I_v}, \mathbf{L_u}, \mathbf{L_v}$.

D. Relations to Existing Methods

The proposed framework is related to various existing methods. In particular, we compare the proposed method with DDP [1] and minimax DDP [2] in this section. The proposed CG-DDP is derived similarly as the original DDP [1]. However, CG-DDP is based on a different problem formulation (7) which leads to two cooperative control policies. Notice that when the eigenvalues of $Q_{\mathbf{vv}}$ are sufficiently large, the CG-DDP policy is equivalent to the standard DDP policy. For a simple example, in the scalar case we have

$$\begin{aligned} &Q_{\mathbf{vv}} \to \infty \Longrightarrow \mathbf{H_v} \to \infty \Longrightarrow \mathbf{I_v}, \mathbf{L_v} \to 0 \Longrightarrow \delta\mathbf{v}_t^* \to 0, \\ &Q_{\mathbf{vv}} \to \infty \Longrightarrow \mathbf{H_u} \to Q_{\mathbf{uu}} \Longrightarrow \mathbf{I_v} \to Q_{\mathbf{uu}}^{-1}Q_{\mathbf{u}}, \mathbf{L_v} \to Q_{\mathbf{uu}}^{-1}Q_{\mathbf{ux}} \\ &\qquad \Longrightarrow \delta\mathbf{u}_t^* \to \underbrace{Q_{\mathbf{uu}}^{-1}Q_{\mathbf{u}} + Q_{\mathbf{uu}}^{-1}Q_{\mathbf{ux}}\delta\mathbf{x}_t}_{\textbf{DDP policy}}. \end{aligned}$$

Intuitively when $\mathbf{v}$ is sufficiently expensive, we retrieve the DDP solution. Therefore CG-DDP can be viewed as a generalization of DDP. Furthermore, according to the group rationality of CSDG policies, CG-DDP yields no worse solution under stochastic disturbances compared to DDP.

One might notice that our approach is related to the minimax DDP [2]. The minimax DDP is derived from a non-cooperative game formula

$$\min_{\mathbf{u}_t}\max_{\mathbf{v}_t}\Big[q(\mathbf{x}_T) + \int_t^T \mathcal{L}\big(t, \mathbf{x}_t, \mathbf{u}_t, \mathbf{v}_t\big)\mathrm{d}t\Big]. \tag{18}$$

While the scheme is different from our case, the resulting policy formulations share some similarities. In particular eq.(11) appears in the minimax-DDP as well. The minimax DDP is based on H^∞ control theory such that the optimal control gain would minimize the effects of the worst disturbance to the system. There are several key differences between CG-DDP and minimax DDP. In particular, in minimax-DDP the non-cooperative policy $\mathbf{v}$ is not applied to the physical systems since it is treated as disturbances. In CG-DDP the policies for both player $\mathbf{u}, \mathbf{v}$ are applied. Furthermore, in CG-DDP the backward Riccati equations are different from the minimax-DDP case. In particular the coupling terms between $\mathbf{u}$ and $\mathbf{v}$ (e.g.,$Q_{\mathbf{uv}}$) and noise-related terms appear in CG-DDP (14). Compared to minimax-DDP, the major benefits of CG-DDP can be summarized as follows: i) in minimax-DDP the existence of solution depends on tuning of the cost function. For instance for a cost defined as $\mathcal{L}(\mathbf{x}_t, \mathbf{u}_t, \mathbf{v}_t) = (\mathbf{x}_t - \mathbf{x}_t^d)^\mathrm{T}\mathbf{Q}(\mathbf{x}_t - \mathbf{x}_t^d) + \mathbf{u}_t^\mathrm{T}\mathbf{R_u}\mathbf{u}_t - \mathbf{v}_t^\mathrm{T}\mathbf{R_v}\mathbf{v}_t$, the existence of the minimax solution depends on the choice of $\mathbf{R_v}$. A related discussion and analysis on the tuning of $\mathbf{R_v}$ for linear systems can be found in [33]. In CG-DDP, the existence of solution does not depend on the tuning of the cost functions. ii) In minimax-DDP the optimal solution is a saddle point while in CG-DDP the optimal solution is an extremum. Numerically it is challenging to find the saddle point since monotonicity during convergence cannot be assured in the Min-Max case where $\mathbf{H_u}$ and $\mathbf{H_v}$ are positive definite and negative definite, respectively. Furthermore, it has been shown [21] that the Min-Max criterion results in longer learning time for RL tasks. We

will compare CG-DDP and minimax-DDP numerically in Sec. V.

IV. Belief Space CG-DDP under Learned Dynamics

We have introduced the general framework of CG-DDP. In this section we introduce a RL algorithm based on probabilistic models learning.

A. Bayesian Nonparametric Model Learning

The goal of this section is to learn the unknown passive dynamics $\mathbf{f}$ as a probabilistic model. We assume the control matrix $\mathbf{B}$ is given or learned a-priori. Estimating $\mathbf{B}$ using simple least squares method is well-understood and has been applied in various robotic control tasks [34]. The function $\mathbf{f}$ can be viewed as an inference with the goal of inferring the uncontrolled transition $\mathrm{d}\tilde{\mathbf{x}}_t = \mathrm{d}\mathbf{x}_t - \mathbf{B}_t\mathbf{u}_t\mathrm{d}t$ given $\mathbf{x}_t$. Given a sequence of sampled states $\{\mathbf{x}_0, \dots \mathbf{x}_T\}$, and the corresponding state transition $\{\mathrm{d}\tilde{\mathbf{x}}_0, \dots, \mathrm{d}\tilde{\mathbf{x}}_T\}$, we adopt the Gaussian process (GP) approach [35, 13] for model learning. Although the stochastic system is non-zero mean, here we still use standard prior of zero-mean in order to demonstrate the robustness of the proposed framework. The covariance is defined via a kernel function

$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \sigma_s^2 \exp(-\frac{1}{2}(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}\mathbf{W}(\mathbf{x}_i - \mathbf{x}_j)) + \sigma_n^2,$$

with $\sigma_s, \sigma_n, \mathbf{W}$ the hyper-parameters. As the standard GP, we assume independent outputs (no correlation between each output dimension). To propagate the GP-based dynamics over a trajectory of time horizon T we employ the moment matching approach [36, 13] to compute the predictive distribution. Given an input distribution over the state $\mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$, the predictive distribution over the passive dynamics is computed as $\mathcal{N}(\boldsymbol{\mu}_{ft}, \boldsymbol{\Sigma}_{ft})$, therefore the predictive distribution over the state at $t + \mathrm{d}t$ can be approximated as a Gaussian $\mathrm{p}(\mathbf{x}_{t+\mathrm{d}t}) \sim \mathcal{N}(\boldsymbol{\mu}_{t+\mathrm{d}t}, \boldsymbol{\Sigma}_{t+\mathrm{d}t})$ such that

$$\begin{aligned} \boldsymbol{\mu}_{t+\mathrm{d}t} &= \boldsymbol{\mu}_t + \boldsymbol{\mu}_{ft} + \mathbf{B}_t\mathbf{u}_t\mathrm{d}t, \\ \boldsymbol{\Sigma}_{t+\mathrm{d}t} &= \boldsymbol{\Sigma}_t + \boldsymbol{\Sigma}_{ft} + \mathbb{COV}[\mathbf{x}_t, \mathrm{d}\tilde{\mathbf{x}}_t] + \mathbb{COV}[\mathrm{d}\tilde{\mathbf{x}}_t, \mathbf{x}_t]. \end{aligned} \tag{19}$$

The above formulation is used to approximate one-step transition probabilities over the trajectory. Details regarding the moment matching method can be found in [36, 13]. We compute all mean and covariance terms analytically. The hyper-parameters $\sigma_s, \mathbf{W}$ are learned by maximizing the log-likelihood of the training outputs given the inputs [35].

B. Trajectory Optimization in Belief Spaces

Given the learned probabilistic dynamics model, we introduce the Gaussian parameterization of state $\mathbf{x}_t$, called belief $\mathbf{b}_t = \big(\boldsymbol{\mu}_t, \mathrm{vec}(\sqrt{\boldsymbol{\Sigma}_t})\big)$. The redundancy in $\mathrm{vec}(\sqrt{\boldsymbol{\Sigma}_t})$ can be eliminated by only using the upper triangular entries of $\sqrt{\boldsymbol{\Sigma}_t}$. We define the belief dynamics as

$$\mathbf{b}_{t+\mathrm{d}t} = \mathbf{F}(\mathbf{b}_t) + \tilde{\mathbf{B}}(\mathbf{u}_t + \mathbf{v}_t), \tag{20}$$

where $\mathbf{F}$ is the learned model and $\tilde{\mathbf{B}} = \left[\begin{smallmatrix} \mathbf{B} \\ 0 \end{smallmatrix}\right]$. Notice that the model uncertainty is incorporated in the belief state. The goal here is to find a local approximation of the belief dynamics (20) around the neighborhood of a nominal trajectory $(\bar{\mathbf{b}}_t, \bar{\mathbf{u}}_t, \bar{\mathbf{v}}_t)$. In RL literature this nominal trajectory is often provided initially as an expert demonstration [4]. In this paper we do not require any a priori information of the optimal trajectory, therefore the initial $(\bar{\mathbf{b}}_t, \bar{\mathbf{u}}_t, \bar{\mathbf{v}}_t)$ is obtained by applying random controls to the physical system. To build a local approximation of the belief dynamics we define the belief variation $\delta\mathbf{b}_t = \mathbf{b}_t - \bar{\mathbf{b}}_t$, the control variations have been defined in Sec.III-A. Therefore the belief dynamics (20) can be linearized around $(\bar{\mathbf{b}}_t, \bar{\mathbf{u}}_t, \bar{\mathbf{v}}_t)$ such that

$$\delta\mathbf{b}_{t+\mathrm{d}t} = \mathbf{A}\delta\mathbf{b}_t + \tilde{\mathbf{B}}\big(\delta\mathbf{u}_t + \delta\mathbf{v}_t\big). \tag{21}$$

$$\text{where} \quad \mathbf{A} = \frac{\partial \mathbf{F}}{\partial \mathbf{b}} = \begin{bmatrix} \frac{\partial \boldsymbol{\mu}_{t+\mathrm{d}t}}{\partial \boldsymbol{\mu}_t} & \frac{\partial \boldsymbol{\mu}_{t+\mathrm{d}t}}{\partial \sqrt{\boldsymbol{\Sigma}_t}} \\ \frac{\partial \sqrt{\boldsymbol{\Sigma}_{t+\mathrm{d}t}}}{\partial \boldsymbol{\mu}_t} & \frac{\partial \sqrt{\boldsymbol{\Sigma}_{t+\mathrm{d}t}}}{\partial \sqrt{\boldsymbol{\Sigma}_t}} \end{bmatrix}. \tag{22}$$

We compute the partial derivatives $\frac{\partial \boldsymbol{\mu}_{t+\mathrm{d}t}}{\partial \boldsymbol{\mu}_t}, \frac{\partial \boldsymbol{\mu}_{t+\mathrm{d}t}}{\partial \sqrt{\boldsymbol{\Sigma}_t}}, \frac{\partial \sqrt{\boldsymbol{\Sigma}_{t+\mathrm{d}t}}}{\partial \boldsymbol{\mu}_t}, \frac{\partial \boldsymbol{\Sigma}_{t+\mathrm{d}t}}{\partial \sqrt{\boldsymbol{\Sigma}_t}}$ analytically. The linearized belief dynamics can be integrated into the proposed CG-DDP framework. The algorithm is briefly summarized as follow: Initially, a GP dynamics model is learned using N sampled stochastic trajectories by applying random controls (Sec.IV-A). Choose one of them as the nominal trajectory $\bar{\mathbf{b}}, \bar{\mathbf{u}}, \bar{\mathbf{v}}$. Next, CG-DDP is applied to the learned belief dynamics (as in **Algorithm** 1) until convergence. The optimized control policy (eq.17) is applied to the physical system to generate the new nominal trajectory for the next optimization loop. Sample another N-1 trajectories using small variations of the controller. Finally after M trials, the task is learned and we obtain the finally optimized control policy and state trajectory.

V. Experiments and Analysis

In this section we provide simulation results and analysis based on two dynamical systems: the cart and double inverted pendulum system and the PUMA-560 robot manipulator. All experiments were performed in MATLAB.

A. Task Descriptions

1) Cart and double inverted pendulum (CDIP): Swinging up the CDIP is challenging because the system is highly nonlinear and under-actuated with 6 state dimensions, 3 degrees of freedom and 1 control input on the cart. The task is to swing-up the two-link pendulum from the initial position (both point down). We set up the experiment using the following parameters: number of sampled trajectories at each trial $N = 4$; terminal time $T = 1$ and time step $\mathrm{d}t = 0.02$. An example of the task is shown in Fig.2a. The disturbance (Fig.2c) is applied to the cart.

2) PUMA-560 robot manipulator: PUMA-560 is a 3D robotic arm that has 12 state dimensions, 6 degrees of freedom with 6 actuators on each joint. The task is to steer the end-effector to the desired position and orientation. We use $N = 2$; $T = 1.0$ and $\mathrm{d}t = 0.02$. An example of the task is shown in Fig.2b. The disturbance (Fig.2c) is applied to each joint of the arm.

B. Methods for comparison

To demonstrate the performance of CG-DDP with both known and learned dynamics. We compare it with DDP [1], minimax-DDP [2] as well as PDDP [19] in a RL setting.

1) DDP: As one of the most efficient trajectory optimization framework, DDP is still widely applied in control of autonomous systems (e.g.,[4]). As we have shown in Sec.III-D, DDP can be viewed as a special case of the proposed CG-DDP.

2) Minimax-DDP: The minimax-DDP [2] is a robust trajectory optimization method. Rooted in the linear H^∞ control theory, the minimax-DDP framework has a non-cooperative game interpretation eq.(18). Compared to DDP, the minimax-DDP is a more conservative approach such that the disturbance is treated as a non-cooperative player. We have described the major similarities and differences between CG-DDP and minimax-DDP in Sec.III-D.

3) PDDP: PDDP performs DDP in Gaussian belief spaces based on GP dynamics models learned similarly as in [13]. It has shown superior data and computational efficiency comparable to the state of the art model-based RL approach [13].

C. Results and Analysis

We perform 2 experiments for each task under non-Gaussian disturbances. See Fig.2c for examples. Experiment $\#1$ and $\#2$ corresponds to the case of known and learned dynamics, respectively.

1) Experiment #1: In this experiment we compare CG-DDP with the standard DDP and minimax-DDP in terms of control performance under stochastic disturbance and computational efficiency. We assume the dynamics models $\mathbf{f}$ and $\mathbf{B}$ are given. For both tasks we sample 100 stochastic trajectories by applying the optimized control policies. Results of trajectory costs are shown in Fig.3. Both CG-DDP and minimax-DDP show superior performance in terms of robustness such that they successfully steer all trajectories to the targets. While in the standard DDP case, a few trajectories diverge from the target due to the stochastic disturbances especially for the underactuated CDIP task. The major reasons for the performance differences between CG-DDP and DDP can be summarized as follow: on one hand, theoretically the group rationality of cooperative players guarantees that the two-controller coalition works better (or at least equal) than one. On the other hand, CG-DDP policies have higher feedback control gains such that the effects of disturbances are weakened. In Fig.4 we show the comparison in terms of number of iterations required for convergence and average computational time per iteration. CG-DDP shows faster convergence in both tasks. In particular, for the highly underactuated CDIP task, CG-DDP outperforms the minimax-DDP significantly. The slightly improved performance of CG-DDP in terms of convergence is due to larger step sizes at each iteration compared to DDP. Theses step sizes correspond to the inverse of Hessian matrices in eq.(16). The minimax-DDP shows slower convergence than the other two methods due to the lack of monotonicity during convergence.

2) Experiment #2: In this experiment we compare CG-DDP with PDDP in terms of robustness to model error. We assume partial knowledge of the dynamics. More precisely the control matrix $\mathbf{B}$ is given. The purpose of this experiment is to show sensitivities of control policies to disturbance as well as model uncertainty. At each trial we sample 100 stochastic trajectories by applying the learned control policies (initialized with random controls). The success rate is defined as the number of trajectories that satisfy $q(\mathbf{x}_T) < 0.25 \times q(\mathbf{x}_0)$. At each trial we use 5 trajectories for model learning. Both CG-DDP and PDDP keep a fixed size of training data. Results and comparison in terms of success rate are shown in Fig.5. For both tasks CG-DDP shows higher success rates therefore less sensitive to model error. Similar to experiment$\#1$, the outperformance can be interpreted from the group rationality of the data-driven CG-DDP control policies. This feature is carried over from trajectory optimization under known dynamics to belief space planning under learned dynamics.

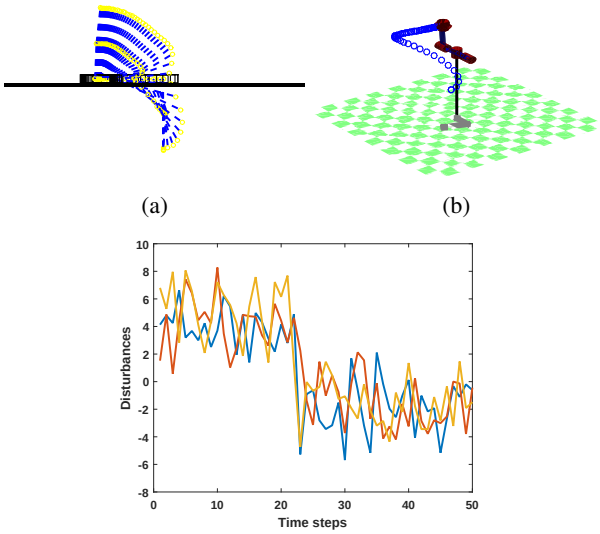

Fig. 2: An example of postures for (a) CDIP and (b) PUMA-560 tasks. (c) Examples of disturbances used in our experiments.

VI. Conclusion and Discussion

While the majority of previous literature on robust control and trajectory optimization utilizes the Min-Max formulation, our work introduces a novel robust trajectory optimization framework based on Cooperative Stochastic Differential Games (CSDG) for nonlinear systems with Poisson and Gaussian disturbances. By proofing the property of group rationality our analysis justifies the use of two controllers for the corresponding CSDG. To address the issue of the *curse of dimensionality* in continuous spaces we derive a novel trajectory optimization algorithm based on Differential Dynamic Programming (DDP). To our knowledge, the proposed algorithm Cooperative Game-DDP (CG-DDP) is the first to

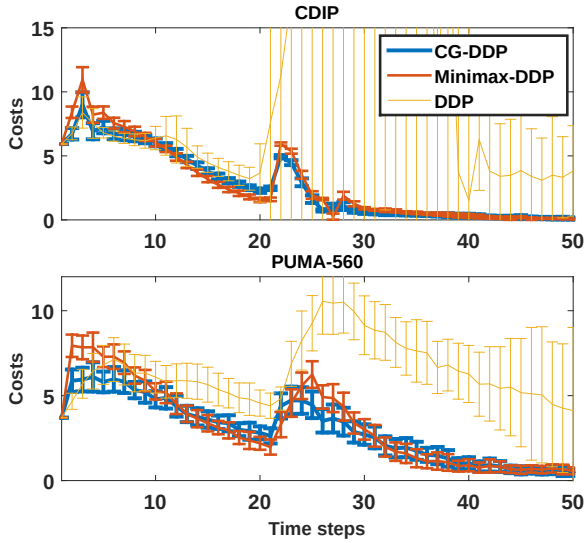

Fig. 3: Experiment #1: State costs of 100 sampled stochastic trajectories by applying the optimized CG-DDP, minimax-DDP and DDP control policies. Solid lines and errorbars show means and standard deviations of sampled trajectoreis. Upper figure: CDIP. Lower figure: PUMA-560.

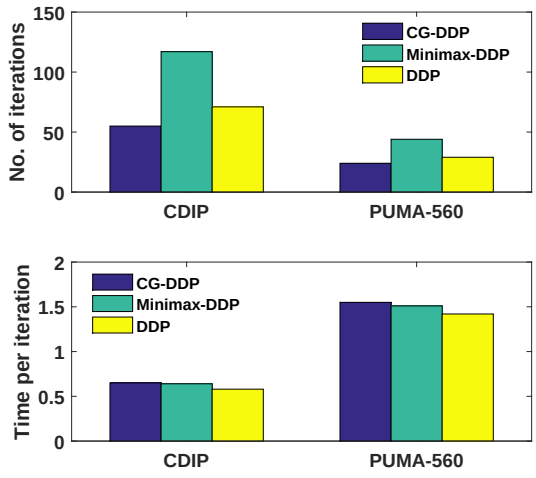

Fig. 4: Experiment #1: For both tasks, comparison of computational efficiency in terms of the number of iteration required to converge (upper figure) and the average computational time (in second) per iteration (lower figure).

numerically solve CSDG for nonlinear systems. CG-DDP combines various attractive characteristics: namely optimality from dynamic programming principle, robustness based on CSDG theory and scalability from local trajectory optimization. Simulation results and comparative analysis demonstrate its performance under external disturbances as well as internal model uncertainties.

Under probabilistic representations of the dynamics, CG-DDP offers robust performance against disturbances under the

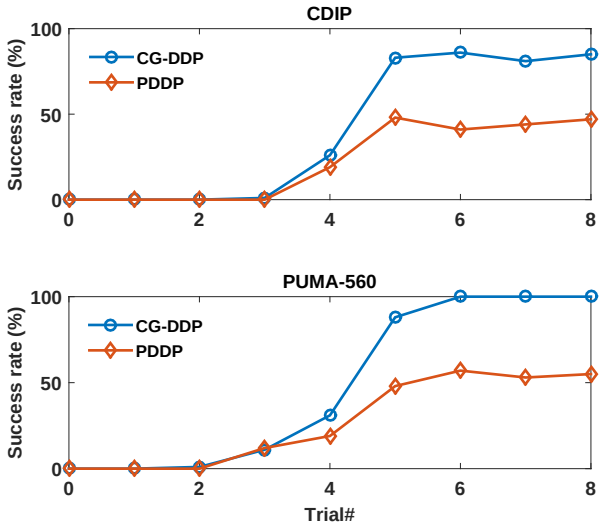

Fig. 5: Experiment #1: comparisons of success rate for the CDIP task (upper figure) and the PUMA-560 task (lower figure).

assumption that the control matrix $\mathbf{B}$ is given. In many robotic systems the control matrix $\mathbf{B}$ corresponds to the inverse of the inertia matrix, which can be identified based on data. A simple least square method with basis functions can be adopted to learn the control matrix for robotic control tasks [34]. While the approach in [34] uses more data to learn the model offline, it can provide a globally accurate estimate of the control transition matrix. When CG-DDP is applied to dynamics with unknown drift, the estimation error of the control transition matrix is essential for the overall performance. This is because CG-DDP provides relatively high gain controls which amplify the estimation error of the control transition matrix.

Future work will focus on extending CG-DDP to the case of systems with entirely unknown dynamics and systems with control limits. Furthermore, applications to real robotic systems will be considered and extensions to partially observable cases will be investigated. This work can be a precursor to several interesting theoretical problems and applicable algorithms in robotics.

REFERENCES

[1] D.H. Jacobson and D.Q Mayne. *Differential dynamic programming*. Elsevier Sci. Publ., 1970.

[2] J. Morimoto and CG Atkeson. Minimax differential dynamic programming: An application to robust biped walking. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1539–1546, 2002.

[3] E. Todorov and W. Li. A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *American Control Conference, 2005*, pages 300–306. IEEE, 2005.

[4] P. Abbeel, A. Coates, M. Quigley, and A. Y Ng. An application of reinforcement learning to aerobatic heli-

copter flight. *Advances in Neural Information Processing Systems (NIPS)*, 19:1, 2007.

[5] Y. Tassa, T. Erez, and W. D. Smart. Receding horizon differential dynamic programming. In *NIPS*, 2007.

[6] J. Van Den Berg, S. Patil, and R. Alterovitz. Motion planning under uncertainty using iterative local optimization in belief space. *The International Journal of Robotics Research*, 31(11):1263–1278, 2012.

[7] Y. Tassa, N. Mansard, and E. Todorov. Control-limited differential dynamic programming. In *2014 IEEE International Conference on Robotics and Automation (ICRA),*, pages 1168–1175. IEEE, 2014.

[8] S. Levine and V. Koltun. Learning complex neural network policies with trajectory optimization. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 829–837, 2014.

[9] S. Levine and P. Abbeel. Learning neural network policies with guided policy search under unknown dynamics. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1071–1079, 2014.

[10] I. Mordatch and E. Todorov. Combining the benefits of function approximation and trajectory optimization. In *Robotics: Science and Systems (RSS)*, 2014.

[11] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. Adaptive computation and machine learning. MIT Press, Cambridge, 1998.

[12] M.P. Deisenroth, G. Neumann, and J. Peters. A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2(1-2):1–142, 2013.

[13] M Deisenroth, D. Fox, and C Rasmussen. Gaussian processes for data-efficient learning in robotics and control. *IEEE Transsactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27:75–90, 2014.

[14] E. Theodorou, J. Buchli, and S. Schaal. A generalized path integral control approach to reinforcement learning. *The Journal of Machine Learning Research*, 11:3137–3181, 2010.

[15] J. Peters, K. Muelling, and Y. Altun. Relative entropy policy search. In *Proceedings of the Twenty-Fourth National Conference on Artificial Intelligence (AAAI), Physically Grounded AI Track*, 2010.

[16] J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural networks*, 21(4):682–697, 2008.

[17] J. Kober and J. Peters. Policy search for motor primitives. In D. Schuurmans, J. Benigio, and D. Koller, editors, *Advances in Neural Information Processing Systems 21 (NIPS 2008)*. Cambridge, MA: MIT Press, 2008.

[18] S. Kuindersma, R. Grupen, and A. Barto. Variational Bayesian optimization for runtime risk-sensitive control. In *Robotics: Science and Systems VIII (RSS)*, Sydney, Australia, July 2012.

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

[20] K. Zhou, J. C. Doyle, K. Glover, et al. *Robust and optimal control*, volume 40. Prentice Hall New Jersey, 1996.

[21] J. Morimoto and K. Doya. Robust reinforcement learning. *Neural computation*, 17(2):335–359, 2005.

[22] H. J. Kappen. Path integrals and symmetry breaking for optimal control theory. *Journal of Statistical Mechanics: Theory and Experiment*, 11:P11011, 2005.

[23] E. Todorov. Eigenfunction approximation methods for linearly-solvable optimal control problems. In *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning, 2009 (ADPRL'09)*, pages 161–168. IEEE, 2009.

[24] E. Todorov. Efficient computation of optimal actions. *Proceedings of the national academy of sciences*, 106(28):11478–11483, 2009.

[25] D.H Jacobson. Optimal stochastic linear systems with exponential performance criteria and their relation to deterministic differential games. *IEEE Transactions on Automatic Control*, 18(2):124–131, 1973.

[26] D.W. Yeung and L.A. Petrosjan. *Cooperative stochastic differential games*. Springer Science & Business Media, 2006.

[27] K. Leyton-Brown and Y. Shoham. Essentials of game theory: A concise multidisciplinary introduction. *Synthesis lectures on artificial intelligence and machine learning*, 2(1):1–88, 2008.

[28] L.A. Petrosyan and N.N. Danilov. Cooperative differential games and their applications. *Izd. Tomskogo University, Tomsk*, 1982.

[29] L.Z Liao and C.A Shoemaker. Convergence in unconstrained discrete-time differential dynamic programming. *Automatic Control, IEEE Transactions on*, 36(6):692–706, 1991.

[30] E.A. Theodorou and E. Todorov. Stochastic optimal control for nonlinear markov jump diffusion processes. In *American Control Conference (ACC), 2012*, pages 1633–1639. IEEE, 2012.

[31] F.B. Hanson. *Applied stochastic processes and control for Jump-diffusions: modeling, analysis, and computation*, volume 13. Siam, 2007.

[32] E. Theodorou, Y. Tassa, and E. Todorov. Stochastic differential dynamic programming. In *American Control Conference (ACC), 2010*, pages 1125–1132. IEEE, 2010.

[33] T.L. Vincent. *Nonlinear and optimal control systems*. John Wiley & Sons, 1997.

[34] K. Kinjo, E. Uchibe, and K. Doya. Evaluation of linearly solvable markov decision process with dynamic model learning in a mobile robot navigation task. *Frontiers in neurorobotics*, 7, 2013.

[35] C.K.I Williams and C.E. Rasmussen. *Gaussian processes for machine learning*. MIT Press, 2006.

[36] J. Quinonero Candela, A. Girard, J. Larsen, and C. E. Rasmussen. Propagation of uncertainty in bayesian kernel models-application to multiple-step ahead forecasting. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2003.