T-PFC: A Trajectory-Optimized Perturbation Feedback Control Approach

Karthikeya S Parunandi[1] and Suman Chakravorty[2]

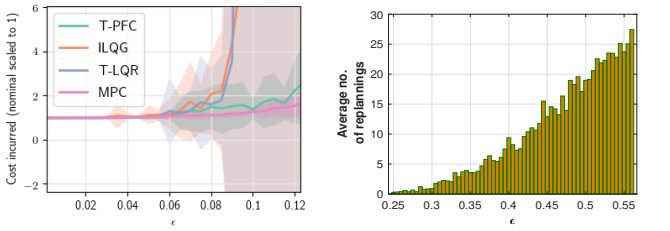
***Abstract*—Traditional stochastic optimal control methods that attempt to obtain an optimal feedback policy for nonlinear systems are computationally intractable. In this paper, we derive a decoupling principle between the open loop plan, and the closed loop feedback gains, that leads to a deterministic perturbation feedback control based solution (T-PFC) to fully observable stochastic optimal control problems, that is near-optimal. Extensive numerical simulations validate the theory, revealing a wide range of applicability, coping with medium levels of noise. The performance is compared against a set of baselines in several difficult robotic planning and control examples that show near identical performance to NMPC while requiring much lesser computational effort.**

***Index Terms*—Motion and Path Planning, Motion Control, Nonholonomic Motion Planning, Optimization and Optimal Control.**

I. INTRODUCTION

STOCHASTIC optimal control is concerned with obtaining control laws under uncertainty, minimizing a user-defined cost function while being compliant with its model and constraints. This problem frequently arises in robotics, where, planning a robot's motion under sensor, actuator and environmental limitations is vital to achieve a commanded task. At present, online planning methods such as Model Predictive Control (MPC) are preferred over offline methods. However, it takes a toll on the onboard computational resources. On the other hand, offline solutions are susceptible to drift, and cannot deal with a dynamic environment. In this paper, we propose a composite approach that merges the merits of both approaches i.e, computation off-line and a robust feedback control online, while re-planning, unlike in MPC, is performed only rarely, and is typically required only beyond moderate levels of noise.

The main contributions of this paper are as follows: (a) to demonstrate the decoupling between the deterministic open-loop and the closed loop feedback control of perturbations, in a fully-observed stochastic optimal setting, that is near-optimal, (b) to propose a novel method based on the aforementioned decoupling principle to deal with robotic stochastic optimal control problem, and (c) to draw comparisons between the proposed approach and the non-linear MPC framework, aimed at re-examining the widespread use of non-linear MPC in robotic planning and control.



(a) Cost comparison (b) No. of re-plannings for $\epsilon > 0.25$

Fig. 1: (a) Cost evolution over a feasible range of ϵ for a car-like robot, where ϵ is a measure of the noise in the system. Note that the performance of T-PFC is close to NMPC for a wide range of noise levels, while T-PFC takes approximately $100\times$ less time to execute (see Table I). (b) No. of re-plannings for above-moderate noise levels in the car-like robot simulation in gazebo using T-PFC is still around 8 times less than NMPC.

II. RELATED WORK

In fully observable systems, decision-making is typically modeled as a Markov Decision Process (MDP). Methods that try to solve MDPs using dynamic programming/HJB face the 'curse of dimensionality' in high-dimensional spaces while discretizing the state space [1]. Hence, most successful/practical methods are based on Pontryagin's maximum principle [2] though it results in locally optimal solutions. Iterative methods such as ILQG [3], DDP [4] and stochastic DDP [5] fall under this category. They expand the optimal cost-to-go and the system dynamics about a nominal, which is updated with every iteration. ILQG relies on the quadratic expansion of the cost-to-go and a linear expansion of system dynamics. DDP/stochastic-DDP considers the quadratic approximation of both. The convergence of these methods is similar to Newton's method. These methods generally optimize the open loop and the linear feedback gain together in an iterative fashion. Differently, in our approach, owing to the decoupling, the open loop optimal control sequence is obtained using a state-of-the-art Nonlinear Programming (NLP) solver, and given this open loop sequence, the optimal feedback gain is obtained using the "decoupled" gain equations. This, in turn, avoids, the expensive recursive Ricatti solutions required by ILQG and DDP techniques (also see Sec. IV).

Manuscript received: February, 25, 2019; Revised May, 14, 2019; Accepted June, 18, 2019.

This paper was recommended for publication by Editor Nancy Amato upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by (organizations/grants which supported the work.)

The authors are with the Department of Aerospace Engineering, Texas A&M University, College Station, TX 77843 USA (e-mail: karthikeyasharma91@gmail.com[1]; schakrav@tamu.edu[2])

Digital Object Identifier (DOI): see top of this page.

Model Predictive Control (MPC) is a popular planning and control framework in robotics. It bypasses the curse of dimensionality by repeatedly generating open-loop controls through the numerical solution of a finite horizon constrained optimal control problem at every discrete time-step [6]. Initially employed in chemical process industry [7], MPC has found widespread application in robotics owing to its ability to handle nonlinearity and constraints. Currently, this framework is well-established in the field and has demonstrated success in diverse range of problems including manipulation [8], visual servoing [8], and motion planning. In robotic motion planning, MPC is widely in use for motion planning of mobile robots, manipulators, humanoids and aerial robots such as quadrotors [9]. Despite its merits, it can be computationally very expensive, especially in context of robot planning and control, since (a) unlike in process industries, typical robotic systems demand re-planning online at high frequency, (b) most systems have highly non-linear dynamical models and (c) constraints apply both on state and controls. Hence, the nonlinear-MPC (NMPC) poses a number of challenges in practical implementation [11]. Lighter variants of MPC such as LMPC, explicit MPC [11], tube-based MPC [12] and other simplified NMPC-based methods [10][11] have emerged. However, LMPC gradually induces uncertainties and fails for highly non-linear systems where the range of linearization is narrow and inadequate [6]. Explicit MPC is not practical for higher state and input states due to expensive memory requirements [11]. In [13], the authors proposed a decoupling principle under a small noise assumption and demonstrated first order near optimality of the decoupled control law for general non-linear systems.

This paper establishes a decoupling principle that consists of a nominal open loop controls sequence along with a precisely defined linear feedback law dependent on the open loop. The latter is derived using a perturbation expansion of the Dynamic Programming equation, that is near optimal to second order, and hence, can work for even moderate noise levels. Further, we perform an extensive empirical comparison of our proposed technique, the "Trajectory-optimized Perturbation Feedback Control (T-PFC)", with the NMPC technique, that shows near identical performance up to moderate noise levels, while taking approximately as much as $100\times$ less time than NMPC to execute in some examples (cf. Fig. 1 and Table I).

III. PROBLEM FORMULATION AND PRELIMINARIES

This section outlines the details of the system considered and the problem statement.

A. System description

Let $\mathbf{x_t} \in \mathcal{X} \subset \mathbb{R}^{n_x}$ and $\mathbf{u_t} \in \mathcal{U} \subset \mathbb{R}^{n_u}$ denote the system state and control input at time t respectively, with $\mathcal{X}$ and $\mathcal{U}$ being corresponding vector spaces. We consider a control-affine nonlinear state propagation model as $\mathbf{x_{t+1}} = f(\mathbf{x_t}) + g(\mathbf{x_t})\mathbf{u_t} + \epsilon\sqrt{\mathbf{dT}}\omega_\mathbf{t}$, where, $\omega_\mathbf{t} \in \mathcal{N}(0, \mathbf{I})$ is an i.i.d. zero mean Gaussian noise with variance $\mathbf{I}$. It is derived from the noiseless continuous model: $\dot{\mathbf{x}}_\mathbf{t} = \bar{f}(\mathbf{x_t}) + \bar{g}(\mathbf{x_t})\mathbf{u_t}$, as follows: Let dT be the discretization time for the continuous time Stochastic Differential Equation (SDE) : $\mathbf{dx} = \bar{f}(\mathbf{x})dT + \bar{g}(\mathbf{x})\mathbf{u}dT + \epsilon\mathbf{dw}$, where ϵ is a scaling factor. The discrete time dynamics are obtained from the SDE as follows: $f(\mathbf{x_t}) = \mathbf{x_t} + \bar{f}(\mathbf{x_t})dT$, $g(\mathbf{x_t}) = \bar{g}(\mathbf{x_t})dT$ and the noise term becomes $\epsilon\sqrt{dT}\omega_\mathbf{t}$, where $\omega_\mathbf{t}$ are standard Normal random variables. The reason we explicitly introduce the discretization time dT will become clear later in this section. It is assumed from hereon that $O(dT^2)$ terms are negligible, i.e, the discretization time is small enough.

B. Stochastic optimal control problem

Given an initial state $\mathbf{x_0}$, the problem of stochastic optimal control [15], for a fully observed system, is to solve

$$\min_{\pi} \mathop{\mathbb{E}}_{\omega_t}\Big[C_N(\mathbf{x_N}) + \sum_{t=0}^{N-1} C_t(\mathbf{x_t}, \mathbf{u_t})\Big] \tag{1}$$
$$s.t \quad \mathbf{x_{t+1}} = f(\mathbf{x_t}) + g(\mathbf{x_t})\mathbf{u_t} + \epsilon\sqrt{dT}\omega_t$$

for a sequence of admissible control policies $\pi = \{\pi_0, \pi_1, ..\pi_t, ., \pi_{N-1}\}$, where $\pi_t : \mathcal{X} \to \mathcal{U}$, $C_t : \mathcal{X} \times \mathcal{U} \to \mathcal{R}$ denotes the incremental cost function and $C_K : \mathcal{X} \to \mathcal{R}$, the terminal cost.

C. Definitions

Let $(\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{u}}_\mathbf{t})$ represent the nominal trajectory of the system, with its state propagation described by the model, $\bar{\mathbf{x}}_\mathbf{t+1} = f(\bar{\mathbf{x}}_\mathbf{t}) + g(\bar{\mathbf{x}}_\mathbf{t})\bar{\mathbf{u}}_\mathbf{t}$. Let $(\delta\mathbf{x_t}, \delta\mathbf{u_t})$ denote the perturbation about its nominal, defined by $\delta\mathbf{x_t} = \mathbf{x_t} - \bar{\mathbf{x}}_\mathbf{t}, \delta\mathbf{u_t} = \mathbf{u_t} - \bar{\mathbf{u}}_\mathbf{t}$. Now, by Taylor's expansion of (1) about the nominal $(\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{u}}_\mathbf{t})$ and the zero mean $\mathbf{w_t}$, the state perturbation can be written as $\delta\mathbf{x_{t+1}} = A_t\delta\mathbf{x_t} + B_t\delta\mathbf{u_t} + \epsilon\sqrt{\mathbf{dT}}\omega_\mathbf{t} + r_t$, where $A_t = \frac{\partial f(\mathbf{x_t})}{\partial \mathbf{x_t}}|_{\bar{\mathbf{x}}_\mathbf{t}} + \frac{\partial g(\mathbf{x_t})}{\partial \mathbf{x_t}}|_{\bar{\mathbf{x}}_\mathbf{t}}\bar{\mathbf{u}}_\mathbf{t}$, $B_t = g(\bar{\mathbf{x}}_\mathbf{t})$ and r_t represents higher order terms.

Let $\bar{J}_t(\mathbf{x_t})$ denote the optimal cost-to-go function at time t from $\mathbf{x_t}$ for the deterministic problem (i.e, $\epsilon = 0$), and $\bar{J}^\epsilon_t(\mathbf{x_t})$ denote the optimal cost-to-go function of the stochastic problem. We expand the deterministic cost-to-go quadratically about the nominal state in terms of state perturbations as $\bar{J}_t(\mathbf{x_t}) = \bar{J}_t(\bar{\mathbf{x}}_\mathbf{t}) + G_t\delta\mathbf{x_t} + \frac{1}{2}\delta\mathbf{x_t}^\intercal P_t\delta\mathbf{x_t} + q_t$, where, $G_t = \frac{\partial \bar{J}_t(\mathbf{x_t})}{\partial \mathbf{x_t}}^\intercal|_{\bar{\mathbf{x}}_\mathbf{t}}$, $P_t = \frac{\partial^2 \bar{J}_t(\mathbf{x_t})}{\partial^2 \mathbf{x_t}}|_{\bar{\mathbf{x}}_\mathbf{t}}$ and q_t denotes the higher order terms.

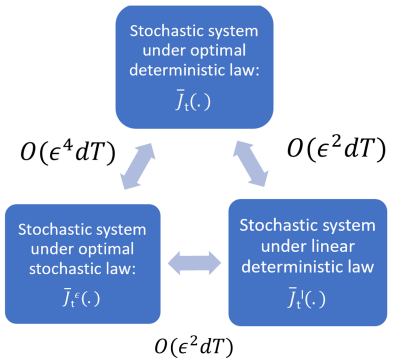
Finally, we consider a step cost function of the form $C_t(\mathbf{x_t}, \mathbf{u_t}) = l(\mathbf{x_t}) + \frac{1}{2}\mathbf{u_t}^\intercal R\mathbf{u_t}$ and let $L_t = \frac{\partial l(\mathbf{x_t})}{\partial \mathbf{x_t}}|_{\bar{\mathbf{x}}_\mathbf{t}}$ and $L_{tt} = \frac{\partial^2 l(\mathbf{x_t})}{\partial^2 \mathbf{x_t}}|_{\bar{\mathbf{x}}_\mathbf{t}}$. Using the definitions above, we assume that the functions $f(\mathbf{x_t})$, $\bar{J}(\mathbf{x_t})$ and $l(\mathbf{x_t})$ are sufficiently smooth over their domains such that the requisite derivatives exist and are uniformly bounded.

IV. A NEAR OPTIMAL DECOUPLING PRINCIPLE

This section states a near-optimal decoupling principle that forms the basis of the T-PFC algorithm presented in the next section. Our program in this section shall be as follows:

- *Decoupling:* First, we shall show that the optimal open loop control sequence of the deterministic problem (given by the gains G_t) can be designed independent of the closed loop gains determined by P_t, i.e, the P_t do not affect the G_t equations for an optimal control sequence in the deterministic problem.
- *Step A:* Next, we shall only keep the first two terms in the optimal deterministic feedback law, i.e., $\mathbf{u_t^l} = \bar{\mathbf{u}}_\mathbf{t} + K_t\delta\mathbf{x_t}$, and show that the closed loop performance of the truncated linear law is within $O(\epsilon^2 dT)$ of the full deterministic feedback law when applied to the *stochastic system*.
- *Step B:* Finally, we will appeal to a result by Fleming [26] that shows that the closed loop performance of the full deterministic law applied to the stochastic system is within $O(\epsilon^4 dT)$ of the optimal stochastic closed loop, and show that the stochastic closed loop performance of the truncated linear feedback law is within $O(\epsilon^2 dT)$ of the optimal stochastic closed loop

The scheme above is encapsulated in Fig. 2.



(a)

Fig. 2: Schematic of the Near-Optimal Decoupling Principle

Proposition 1: Decoupling. Given an optimal nominal trajectory, the backward evolutions of the deterministic gain G_t and the covariance P_t of the optimal cost-to-go function $\bar{J}_t(\mathbf{x_t})$, initiated with $G_N = \frac{\partial \bar{C}_N(\bar{x}_N)}{\partial \mathbf{x_N}}^\intercal|_{\bar{\mathbf{x}}_\mathbf{N}}$ and $P_N = \frac{\partial^2 \bar{C}_N(\bar{x}_N)}{\partial^2 \mathbf{x_N}}|_{\bar{\mathbf{x}}_\mathbf{N}}$ respectively, are as follows:

$$G_t = L_t + G_{t+1}A_t \tag{2}$$

$$P_t = L_{tt} + A_t^\intercal P_{t+1}A_t - K_t^\intercal S_t K_t + G_{t+1} \otimes \tilde{R}_{t,xx} \tag{3}$$

for $t = \{0, 1, ..., N-1\}$, where, $S_t = (R_t + B_t^\intercal P_{t+1}B_t)$, $K_t = -S_t^{-1}(B_t^\intercal P_{t+1}A_t + (G_{t+1} \otimes \tilde{R}_{t_{xu}})^\intercal)$, $\tilde{R}_{t,xx} = \nabla^2_{xx} f(\mathbf{x_t})|_{\bar{\mathbf{x}}_\mathbf{t}} + \nabla^2_{xx} g(\mathbf{x_t})|_{\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{u}}_\mathbf{t}}$, $\tilde{R}_{t,xu} = \nabla^2_{xu}(f(\mathbf{x_t}) + g(\mathbf{x_t})\mathbf{u_t})|_{\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{u}}_\mathbf{t}}$ where ∇^2_{xx} represents the Hessian of a vector-valued function w.r.t x and $\otimes$ denotes the tensor product.

Proof for the above is provided in the appendix section. In essence, the key step in the proof of proposition-1 is in realizing that when the nominal trajectory is optimal, the term corresponding to the open-loop control trajectory vanishes in deriving an expression for perturbed control as shown in equation (4) and thereafter. This means that the dependency of the perturbed variables in the design of the nominal trajectory is nullified resulting in equations (2) and (3). It may be noted here that equation (2) corresponds to the co-state equation following the first order optimality conditions over an optimal nominal trajectory, whereas equation (3) is a discrete time dynamic Riccati-like equation dictating the feedback law design. The consequence of the above result is that the second order sensitivity matrix in the expansion of the cost, P_t which determines the feedback gain K_t, doesn't influence the first order sensitivity matrix G_t (the co-state) that determines the optimal open-loop sequence. Thus, the decoupling between the nominal and linear feedback holds true. In other words, the design of an optimal control policy in a fully-observed problem as in (1) can be decoupled into the design of an open-loop deterministic nominal $(\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{u}}_\mathbf{t})$ and then a linear feedback law whose coefficients can be extracted through a time-recursive propagation of (2) and (3), but which is nonetheless near optimal to second order $(O(\epsilon^2 dT))$ as we shall show below.

Step A. Let the optimal deterministic feedback law for the deterministic system $(\epsilon = 0)$ be given by: $\mathbf{u_t}(\mathbf{x_t}) = \bar{\mathbf{u}}_\mathbf{t} + K_t\delta\mathbf{x_t} + R(\delta\mathbf{x_t})$. The result above gives us the recursive equations required to solve for $\bar{\mathbf{u}}_\mathbf{t}$ in terms of G_t, and K_t in terms of P_t. Consider the truncated linear feedback law, i.e., $\mathbf{u_t^l}(\mathbf{x_t}) = \bar{\mathbf{u}}_\mathbf{t} + K_t\delta\mathbf{x_t}$. Now, we shall apply the control laws $\mathbf{u_t}(.)$ and $\mathbf{u_t^l}(.)$ to the stochastic system $(\epsilon \neq 0)$ and compare the closed loop performance. It can be shown that the state perturbations from the nominal under the optimal deterministic law evolve according to $\delta\mathbf{x_{t+1}} = \bar{A}_t\delta\mathbf{x_t} + B_t R(\delta\mathbf{x_t}) + S_t(\delta\mathbf{x_t}) + \epsilon\sqrt{dT}\omega_t$, while that under the truncated linear law evolves according to $\delta\mathbf{x_{t+1}^l} = \bar{A}_t\delta\mathbf{x_t^l} + S_t(\delta\mathbf{x_t^l}) + \epsilon\sqrt{dT}\omega_t$, where $\bar{A}_t = A_t + B_t K_t$ is the linear closed loop part, and $S_t(.)$ are the second and higher order terms in the dynamics. The closed loop cost-to-go under the full deterministic feedback law is then given by: $\bar{J}_k(\mathbf{x_k}) = E[\sum_{t=k}^N c(\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{u}}_\mathbf{t}) + C_t^1\delta\mathbf{x_t} + H_t(\delta\mathbf{x_t})]$, and that for the truncated linear law is given by: $\bar{J}_k^l(\mathbf{x_k}) = E[\sum_{t=k}^N c(\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{u}}_\mathbf{t}) + C_t^1\delta\mathbf{x_t^l} + H_t(\delta\mathbf{x_t^l})]$, where C_t^1 is the first order coefficient of the step cost expansion that depend only on the nominal $(\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{u}}_\mathbf{t})$, and $H_t(.)$ denote second and higher order terms of the expansions. Then $\bar{J}_k(\mathbf{x_k}) - \bar{J}_k^l(\mathbf{x_k}) = \sum_{t=k}^N \underbrace{E[C_t^1(\delta\mathbf{x_t} - \delta\mathbf{x_t^l})]}_{T_1} + \underbrace{\sum_{t=k}^N E[H_t(\delta\mathbf{x_t}) - H_t(\delta\mathbf{x_t^l})]}_{T_2}$. Consider the deviation between the two closed loops $\delta\mathbf{x_t} - \delta\mathbf{x_t^l} = \bar{A}_t(\delta\mathbf{x_t} - \delta\mathbf{x_t^l}) + B_t R_t(\delta\mathbf{x_t}) + S_t(\delta\mathbf{x_t}) - S_t(\delta\mathbf{x_t^l})$, where note that $||R_t(\delta\mathbf{x_t})|| = O(\epsilon^2 dT)$, as are $||S_t(\delta\mathbf{x_t^l})||$ and $||S_t(\delta\mathbf{x_t})||$ since they consist of second and higher order terms in the feedback law and the dynamics respectively, when $\epsilon\sqrt{dT}$ is small. Therefore, it follows that the closed loop state deviation between the full deterministic and the truncated linear law is $||\delta\mathbf{x_t} - \delta\mathbf{x_t^l}|| = O(\epsilon^2 dT)$. Further, it is also true that $\delta\mathbf{x_t}$ and $\delta\mathbf{x_t^l}$ are both $O(\epsilon\sqrt{dT})$. Hence, using the above it follows that terms $T_1 + T_2$ is $O(\epsilon^2 dT)$. *Therefore, it follows that the difference in the closed loop performance of the full deterministic feedback law and the truncated linear feedback law is* $|\bar{J}_k(\mathbf{x_k}) - \bar{J}_k^l(\mathbf{x_k})| = O(\epsilon^2 dT)$.

Step B: Now, we shall establish the closeness of the optimal stochastic closed loop and the stochastic closed loop

under the truncated linear feedback law. First, we recount a seminal result due to Fleming [26] regarding the "goodness" of the deterministic feedback law for the stochastic system. Fleming considered the continuous time SDE: $d\mathbf{x} = \bar{f}(\mathbf{x})dt + g(\mathbf{x})\mathbf{u}dt + \epsilon d\mathbf{w}$. Let the cost-to-go of the optimal stochastic closed loop be given by $\bar{J}^{\epsilon}(t, \mathbf{x})$, and let the cost-to-go of the closed loop under the deterministic law be given by $\bar{J}(t, \mathbf{x})$. Then, it is shown that the functions $\bar{J}^{\epsilon}$ and $\bar{J}$ have the following perturbation expansion in terms of ϵ: $\bar{J}^{\epsilon} = \varphi + \epsilon^2\theta + \epsilon^4\chi$,and $\bar{J} = \varphi + \epsilon^2\theta + \epsilon^4\chi'$, where φ, θ,χ and χ' are functions of $(t, \mathbf{x})$. Therefore, it follows that the difference in the closed loop performance between the optimal stochastic and optimal deterministic law *when applied to the stochastic system* is $O(\epsilon^4)$!

If we adapt this result to our discrete time case with a time discretization dT, where $O(dT^2)$ is negligible, then the difference between the true stochastic closed loop performance and that under the deterministic optimal law, $|J^{\epsilon}_t(\mathbf{x_t}) - J_t(\mathbf{x_t})| = O(\epsilon^4 dT)$. Thus, using the above result and the result form step A, it follows that *difference between the closed loop performance of the truncated linear feedback law and that of the optimal stochastic closed loop,* $|J^{\epsilon}_t(\mathbf{x_t}) - J^{l}_t(\mathbf{x_T})| = O(\epsilon^2 dT)$ *at the least.* This establishes the near optimality of the truncated linear feedback loop.

ILQG/DDP: The condition in (2) is precisely when the ILQG/ DDP algorithms are deemed to have converged. However, that does not imply that the feedback gain at that stage for ILQG/ DDP is the same as that in Eq. (3), and in fact, the feedback gains of ILQG/ DDP are different from that in Eq. 3 as we shall see in our examples. The basic idea in the development above is to design an open loop optimal sequence, and then design a feedback gain according to Eq. 3, it is in this second step that we differ from ILQG/ DDP (which are methods to get open loop optimal sequences and make no claims about the feedback gains).

V. TRAJECTORY-OPTIMIZED PERTURBATION FEEDBACK CONTROL (T-PFC)

In this section, we formalize the Trajectory-optimized Perturbation Feedback Control (T-PFC) method based on the decoupling principle of the previous section.

A. Nominal Trajectory Design

The optimal nominal trajectory can be designed by solving the deterministic equivalent of problem (1), which can be formulated as an open-loop optimization problem as follows:

$$\min_{\check{\mathbf{u}}} \Big[C_N(\mathbf{x_N}) + \sum_{t=0}^{N-1} C_t(\mathbf{x_t}, \mathbf{u_t})\Big]$$
$$s.t \quad \mathbf{x_{t+1}} = f(\mathbf{x_t}) + g(\mathbf{x_t})\mathbf{u_t}$$

where, $\check{\mathbf{u}} = \{\mathbf{u_0}, \mathbf{u_1}, ..\mathbf{u_{N-1}}\}$. This is a design problem that can be solved by a standard NLP solver. The resultant open-loop control sequence together with a sequence of states obtained through a forward simulation of noise-free dynamics constitute the nominal trajectory.

Constraints on the state and the control can be incorporated in the above problem as follows:

State constraints: Non-convex state constraints such as obstacle avoidance can be dealt by imposing exponential penalty cost as barrier functions. Obstacles can be circumscribed by Minimum Volume Enclosing Ellipsoids (MVEE)[14] that enclose a polygon given its vertices. Such kind of barrier functions can be formulated by [16]: $C_{obs}(\mathbf{x_t}) = \sum_{m=1}^{n} \Gamma_m \exp(-\rho_m(\mathbf{x_t} - c^m)^{\intercal}\mathcal{E}^m(\mathbf{x_t} - c^m))$, where, c^m and $\mathcal{E}$ correspond to the center and geometric shape parameters of the m^{th} ellipsoid respectively. Γ_m and ρ_m are the scaling factors. Obstacles are assimilated into the problem by adding $C_{obs}(\mathbf{x_t})$ to the incremental cost $C_t(\mathbf{x_t}, \mathbf{u_t})$.

Control bounds: Control bounds can safely be incorporated while designing the optimal nominal trajectory as hard constraints in the NLP solver. In this case, the constraints are linear in control inputs and the modified incremental cost function can be written as $C'_t(\mathbf{x_t}, \mathbf{u_t}) = C_t(\mathbf{x_t}, \mathbf{u_t}) + \mu_t(F_t\mathbf{u_t} + H_t)$. The first order condition (4) is then modified to $R_t\bar{\mathbf{u}}_\mathbf{t} + B_t^{\intercal}G_{t+1}^{\intercal} + F_t^{\intercal}\mu_t^{\intercal} = 0$ using KKT conditions [17], which upon utilizing in the derivation of expression for $\delta\mathbf{u_t}$ nullifies the influence of μ_t. Hence, equations (3), (4) and (6) will remain the same.

B. Linear Feedback Controller Design

Given a nominal trajectory $(\bar{\mathbf{x}}, \bar{\mathbf{u}})$, a linear perturbation feedback controller around it is designed by pre-computing the feedback gains. The sequence of K_t is determined by a backward pass of G_t and P_t as described by (3) and (4). The linear feedback control input is given by $\delta\mathbf{u_t} = K_t\delta\mathbf{x_t}$. Hence, $\mathbf{u_t} = \bar{\mathbf{u}}_\mathbf{t} + \delta\mathbf{u_t} = \bar{\mathbf{u}}_\mathbf{t} + K_t(\mathbf{x_t} - \bar{\mathbf{x}}_\mathbf{t})$ forms the near-optimal online control policy. Algorithm-1 outlines the complete T-PFC algorithm.

Re-planning: At any point of time during the execution, if the cost deviates beyond a threshold from the nominal cost $i.e, C_{Th}$, a re-planning can be initiated.

VI. EXAMPLE APPLICATIONS

This section demonstrates T-PFC in simulation with three examples. The `Gazebo` [18] robotics simulator is used as a simulation platform in interface with `ROS` middleware [19]. Numerical optimization is performed using the `Casadi` [20] framework employing the `Ipopt` [21] NLP software. A feasible trajectory generated by the non-holonomic version of the RRT algorithm [22] is fed into the optimizer for an initial guess. Simulations are carried out in a computer equipped with an Intel Core i7 2.80GHz octa-core processor. The results presented in each example are averaged from a set of 100 Monte Carlo simulations for a range of tolerable noise levels ϵ. The proposed approach has been implemented to the problem of motion planning under process noise in the dynamical model to obtain the cost plots and then simulated in a physics engine on a realistic robot model for further analysis.

Noise characterization: Process noise is modeled as a standard Brownian noise added to the system model with a standard deviation of $\epsilon\sqrt{dt}$. Since it is assumed to be additive Gaussian and i.i.d. (even w.r.t. the noise in other state variables), it could account for various kinds of uncertainties

Algorithm 1: T-PFC

Input: Initial State - $\mathbf{x_0}$, Goal State - $\mathbf{x_f}$, Time-step Δt, Horizon - N, System and environment parameters - $\mathcal{P}$;
$t \leftarrow 0$;
`/* Run until the current state is in` ε `proximity to the goal */`
while $\|\mathbf{x_t} - \mathbf{x_f}\| > \varepsilon$ **do**
  `/* Plan at` $t = 0$ `and re-plan when the cost deviation exceeds a threshold or if not within the goal proximity at` $t = N - 1$`. */`
  if *t == 0* or *Cost fraction* $> C_{Th}$ or *t == N-1* **then**
    `/* Open-loop sequence */`
    $(\bar{\mathbf{x}}_{\mathbf{t:N-1}}, \bar{\mathbf{u}}_{\mathbf{t:N-1}}) \leftarrow$Plan$(\mathbf{x_t}, \mathcal{P}, \mathbf{x_f})$
    `/* Closed-loop parameters */`
    Compute parameters:$\{P_{t:N-1}, G_{t:N-1}, K_{t:N-1}\}$
  end if
  Policy evaluation: $\mathbf{u_t} \leftarrow \bar{\mathbf{u}}_\mathbf{t} + K_t(\mathbf{x_t} - \bar{\mathbf{x}}_\mathbf{t})$
  Process: $\mathbf{x_{t+1}} \leftarrow f(\mathbf{x_t}) + g(\mathbf{x_t})\mathbf{u_t} + \epsilon\omega_\mathbf{t}$
  $t \leftarrow t + 1$
end while

including that of parametric, model and the actuator. ϵ is a scaling parameter that is varied to analyze the influence of the magnitude of the noise. Other case-specific parameters are provided in Table II.

For simulation, we use realistic physical robot models in a physics engine in an obstacle-filled environment along with moderate contact friction ($\mu = 0.9$) and drag, which are unaccounted for in our system model. Apart from this model uncertainty, we also introduce actuator noise through an additive Gaussian of standard deviation $\epsilon\sigma_t$, where σ_t is $\|\mathbf{u_s}\|_\infty$.

A. Car-like robot

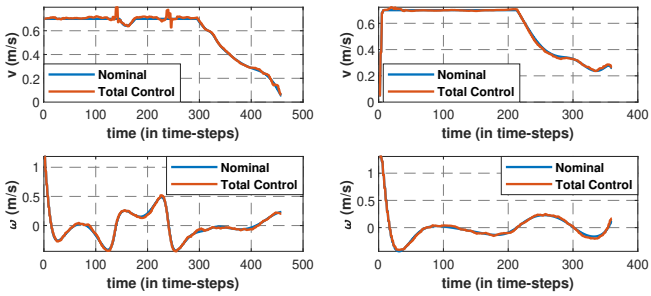
A 4-D model of a car-like robot with its state described by $(x_t, y_t, \theta_t, \phi_t)^\intercal$ is considered. For a control input constituting of the driving and the steering angular velocities, $(u_t, w_t)^\intercal$, the state propagation model is as follows:

$$\dot{x} = u\ cos(\theta), \quad \dot{\theta} = \frac{u}{L}\ tan(\phi)$$
$$\dot{y} = u\ sin(\theta), \quad \dot{\phi} = \omega$$

Fig. 4 shows an example path taken by a car-like robot in an environment filled with 8 obstacles enclosed in MVEEs. Plots in Fig. 3 (a) indicate the averaged magnitude of both the nominal and the total control signals at $\epsilon = 0.25$. The standard deviation of the averaged total control sequence, in both plots, from the nominal is less than one percent of it.

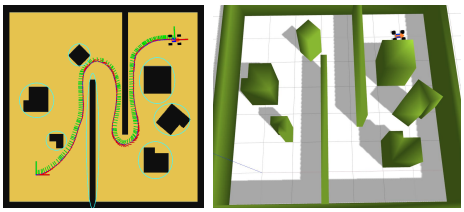
B. Car-like robot with trailers

With n trailers attached to a car-like robot, the state of a car-like-robot is augmented by n dimensions, each additional entry describing the heading angle of the corresponding trailer.



(a) (b)

Fig. 3: Optimal nominal and total control inputs (averaged) at ϵ = 0.25 for (a) a car-like robot and (b) car with trailers



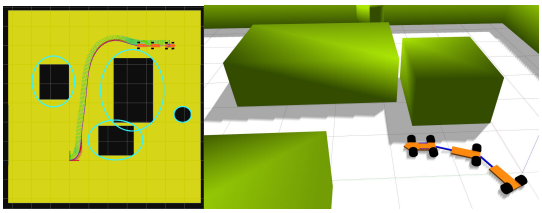
(a) Rviz trajectory (b) Robot's world in Gazebo

Fig. 4: Motion Planning of a car-like robot using T-PFC for an additive control noise of standard deviation = 25% of the norm of saturation controls *i.e*, $\epsilon = 0.25$. The axes along and perpendicular to the robot's trajectory are indicated in red and green colors respectively.

In the current example, n = 2 trailers are considered and their heading angles are given by [23]:

$$\dot{\theta}_1 = \frac{u}{L} sin(\theta - \theta_1)$$
$$\dot{\theta}_2 = \frac{u}{L} cos(\theta - \theta_1) sin(\theta_1 - \theta_2)$$

Hence, the robot has six degrees of freedom. Its world is considered to be composed of four obstacles as shown in Fig. 5. The robot, its environment and its trajectory shown are at $\epsilon = 0.25$. Fig. 5(b) displays the nominal and the total control signals averaged at $\epsilon = 0.25$.



(a) Rviz trajectory (b) Robot's world in Gazebo

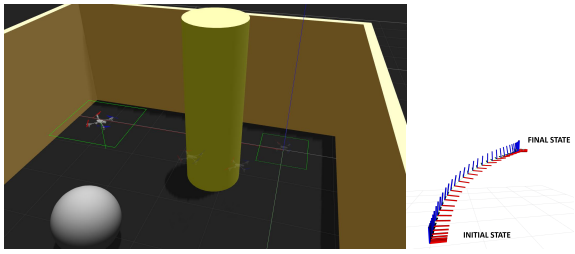
Fig. 5: Motion planning of a car with trailers using T-PFC for an additive control noise of standard deviation set to 25% of the norm of saturation controls *i.e*, $\epsilon = 0.25$. The axes along and perpendicular to the robot's trajectory are indicated in red and green colors respectively.

C. 3D Quadrotor

The 12 DOF state of a Quadrotor comprises of its position, orientation and corresponding rates - $(\mathbf{x_t}, \theta_\mathbf{t}, \mathbf{v_t}, \omega_\mathbf{t})^\intercal$. Forces and torques in its body frame are external inputs in the equations below. However, in the real world (and also in `Gazebo` simulation shown here) the control input is typically fed into the motors. Hence, we consider rotor velocities as the control input, which can be obtained by a linear transformation of forces and torques in body frame. The state propagation model is then given by the following equations [27]:

$$\dot{\mathbf{x}}_\mathbf{t} = \mathbf{v_t}, \qquad \dot{\mathbf{v}}_\mathbf{t} = \mathbf{g} + \frac{1}{m}(R_{\theta_t}\mathbf{F_b} - k_d\mathbf{v_t})$$
$$\dot{\theta}_\mathbf{t} = J_w^{-1}\omega_\mathbf{t}, \qquad \dot{\omega}_\mathbf{t} = I_c^{-1}(\tau_\mathbf{t} - \omega \times I_c\omega)$$

Simulations are performed using an AR.drone model [28] in an environment containing a cylindrical obstacle as shown in Fig. 6.



(a) (b)

Fig. 6: (a) Quadrotor's world in Gazebo - green boxes represent its initial and final positions respectively. (b) Example trajectory in rviz

TABLE I: Average run-time of algorithms in seconds

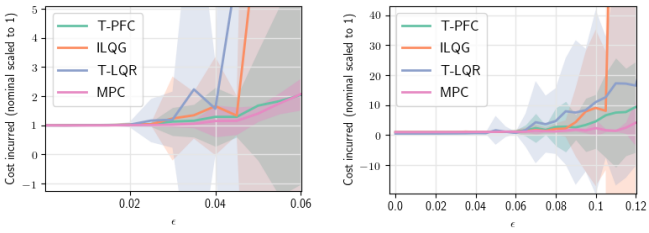
Robot type	MPC	T-LQR	ILQG	T-PFC
Car-like	447.89	4.48	161	4.52
Car with trailers	384.42	4.11	146	4.24
Quadrotor	71	3.33	49	3.5

TABLE II: Simulation parameters

	Car-like	Car with trailers	Quadrotor
$\mathbf{x_0}$	$(0,0,0,0)^\intercal$	$(0,0,0,0,0,0)^\intercal$	$(0,0,0.08,0,$ $0,0,0,0,0,0,0,0)'$
$\mathbf{x_f}$	$(5,5,0,0)^\intercal$	$(5,6,0,0,0,0)^\intercal$	$(2.6,2.4,1.75,$ $0,0,0)^\intercal$
$N, \Delta t$	229, 0.1s	180, 0.1s	60, 0.1s
Control bounds	$\mathbf{u_s^1} = (0.7, -0.7)$ $\mathbf{u_s^2} = (-1.3, 1.3)$	$\mathbf{u_s^1} = (0.7, -0.7)$ $\mathbf{u_s^2} = (-1.3, 1.3)$	$\mathbf{u_s^1} = (20, -20)$ $\mathbf{u_s^i} = (1, -1)$ $i = 2, 3, 4$

VII. DISCUSSION AND COMPARISON OF METHODS

This section empirically details the implications of the decoupling principle and the T-PFC from the examples in the previous section. Further, we make a comparison here with the Non-linear MPC (NMPC) [24], T-LQR [13] and ILQG [3]. Average cost incurred, rate of re-planning and time-taken for an execution are chosen as the performance criteria.



(a) (b)

Fig. 7: Cost evolution over a feasible range of ϵ for (a) car with trailers robot and (b) 3D Quadrotor.

Nonlinear MPC: A deterministic NMPC is implemented with a typical OCP formulation, by re-solving it at every time-step. The NMPC variant implemented here is summarized in Algorithm-2. The prediction horizon is taken as $N-i$ at the i^{th} time-step. In other words, planning is performed all the way till the end rather than for next few time-steps as in typical MPC. This is done for two reasons:
(1) The control sequence obtained this way is equivalent to the deterministic optimal control law that includes higher order terms of feedback control. We wish to juxtapose it with T-PFC that only has a linear feedback (first-order).
(2) Due to high penalty cost of multiple barrier functions, the optimizer is prone to failures with smaller prediction horizons. Also, by the above arrangement, it follows from Bellman's Principle of Optimality that the predicted open-loop control input will be equal to the optimal feedback policy [24]. Therefore, this also results in nominal stability.

Algorithm 2: NMPC

Input: Initial State - $\mathbf{x_0}$, Goal State - $\mathbf{x_f}$, Horizon - N, System and environment parameters - $\mathcal{P}$;
$t \leftarrow 0$;
while $t < N$ **do**
  $(\bar{\mathbf{x}}_{\mathbf{t:N-1}}, \bar{\mathbf{u}}_{\mathbf{t:N-1}}) \leftarrow \text{Plan}(\mathbf{x_t}, \mathbf{u_t}, N-t, \mathbf{x_f}, \mathcal{P})$;
  Process: $\mathbf{x_{t+1}} \leftarrow f(\mathbf{x_t}) + g(\mathbf{x_t})\bar{\mathbf{u}}_\mathbf{t} + \epsilon\omega_\mathbf{t}$
  $t \leftarrow t+1$;
end while

T-LQR: T-LQR is implemented using the same nominal cost as T-PFC. However, the cost parameters of the LQR are tuned entirely separately from the nominal cost [13].
ILQG: ILQG is initiated with the same initial guess as the above three methods. Since the cost contains exponential terms from the barrier functions, it is crucial to carefully choose right parameters for regularization and line search. Regularization is performed by penalizing state deviations in a quadratic modification schedule and an improved line search, both as mentioned in [25]. The feedback gains computed at the final iteration is used for feedback control against noise on top of the resulting open-loop trajectory.
Comparison: From Fig. 1 and 7, the average cost incurred for the systems in each simulation via T-PFC is close to that incurred through an NMPC approach. In other words,

the cost accumulated by our perturbation linear feedback approach is nearly the same as that accumulated by an optimal deterministic control law over the feasible range of ϵ for T-PFC. T-LQR being based on the first order cost approximation, the cost rapidly diverges with increase in the noise level as reflected in Figs. 1 and 6. On the other hand, as ILQG doesn't make any claims regarding feedback, it is expected and is also clear from the same plots that the performance deteriorates rapidly with noise.

Table I shows the average time taken to execute an episode with each of the algorithms with no intermediate re-planning. The total execution time taken by NMPC is nearly 100 times the T-PFC in the most complex of the examples considered. The low online computational demand of T-PFC makes it scalable to implement in systems with higher dimensional state-space.

Another challenging aspect in the implementation of NMPC is generating initial guesses for online optimization. With a number of obstacle constraints or barrier functions, the NMPC optimizer fails to converge to a solution with trivial initializations and even with warm-starting, more so at higher noise levels. In contrast, T-PFC typically solves the optimization problem only once and hence, a one-time initialization is sufficient for the execution. Fig. 1 (b) shows the average rate of re-planning for example-1. Until $\epsilon = 0.25$, no re-planning was necessary in the example of a car-like robot. From Fig. 1 (b), it is evident that even at above-moderate levels of noise, the re-planning frequency is still eight times lesser than that required for an NMPC.

Unlike T-LQR, T-PFC also handles the residual second order terms of cost-to-go as well as system dynamics. This way, tuning is also bypassed as the feedback adapts itself according to the nominal cost. In contrast, T-LQR can apply aggressive controls during feedback depending on LQR parameter-tuning. T-PFC in an attempt to reduce the overall cost, generates smooth and small controls relative to its nominal. This is noticeable in Fig. 3. Also, this fact plays an advantage when the nominal control is on the constraint boundary and it is undesirable for the perturbation control to deviate significantly from the nominal.

The advantage of decoupling between the optimal nominal and the perturbation feedback law is clear when compared with ILQG. Parameter tuning in ILQG for regularization and line-search involves trial and error regulation and is often time consuming to searching for the right set of parameters to every given system, especially when the cost function is non-quadratic and non-polynomial. On the other hand, an NLP solver (using, say, interior-point methods) can be conveniently used in a black box fashion in perturbation feedback approaches such as T-PFC (or even T-LQR) without needing any fine-tuning to result in a deterministic control policy.

Small noise assumption: Though the theory is valid for small noise cases $i.e$, for small epsilons, empirical results suggest a greater range of stability i.e, stability holds even for moderate levels of noise. As long as the noise falls in this range, a precise knowledge of the magnitude of noise is irrelevant as T-PFC is insensitive to noise levels.

Limitations: 1) T-PFC assumes a control-affine system and the cost to be in a specific form. Though many robotic systems are affine in controls, methods like T-LQR have an edge by considering a general nonlinear system.

2) Though T-LQR does not fare well on the cost incurred, it offers a flexibility to tune the feedback parameters according to ones needs, even if that means sacrificing the optimality.

Is deterministic NMPC necessary? Since the MPC framework is broad and there are several ad-hoc techniques that could efficiently solve NMPC, answering this question requires a much deeper analysis. However, our central observation is that the T-PFC (and even T-LQR) method has near identical performance with deterministic NMPC in problems that mandate long horizons. They are also orders of magnitude computationally efficient, both according to the decoupling theory, as well as empirically, based on the problems that we have considered here. In such cases, why not use perturbation feedback techniques instead of NMPC at least until the noise levels predicate frequent re-planning?

VIII. CONCLUSION

In this paper, we have established that in a fully-observed scenario, a deterministic action policy can be split into an optimal nominal sequence and a feedback that tracks the nominal in order to maintain the cost within a tube around the nominal. T-PFC maintains low cost, has low online computation and hence, faster execution. This makes our approach tractable in systems with higher dimensional states. Like MPC, the nominal trajectory design of T-PFC also allows for the inclusion of constraints as described. We have empirically shown that the overall control signals are very close to the saturation boundary, if not with-in, when the nominal is at saturation. Also, T-PFC works with minimal number of re-plannings even at medium noise levels, as against to the traditional principle of deterministic MPC to re-plan in a recurrent fashion irrespective of noise levels. Future work involves exploring this idea of decoupling to partially-observed systems and dealing with nonlinear hard inequality constraints.

APPENDIX

Proof of Proposition 1:

$$\bar{J}_t(\mathbf{x_t}) = \min_{\mathbf{u_t}} J_t(\mathbf{x_t}, \mathbf{u_t}) = \min_{\mathbf{u_t}} \{C_t(\mathbf{x_t}, \mathbf{u_t}) + \bar{J}_{t+1}(\mathbf{x_{t+1}})\}$$

By Taylor's expansion about the nominal state at time $t+1$,

$$\begin{aligned}\bar{J}_{t+1}(\mathbf{x_{t+1}}) =& \bar{J}_{t+1}(\bar{\mathbf{x}}_{\mathbf{t+1}}) + G_{t+1}\delta\mathbf{x_{t+1}} \\ &+ \frac{1}{2}\delta\mathbf{x_{t+1}}^{\intercal} P_{t+1}\delta\mathbf{x_{t+1}} + q_{t+1}(\delta\mathbf{x_{t+1}}).\end{aligned}$$

Substituting $\delta\mathbf{x_{t+1}} = A_t\delta\mathbf{x_t} + B_t\delta\mathbf{u_t} + r_t(\delta\mathbf{x_t}, \delta\mathbf{u_t})$ in the above expansion,

$$\begin{aligned}&\bar{J}_{t+1}(\mathbf{x_{t+1}}) = \bar{J}_{t+1}(\bar{\mathbf{x}}_{\mathbf{t+1}}) + G_{t+1}(A_t\delta\mathbf{x_t} + B_t\delta\mathbf{u_t} + r_t(\delta\mathbf{x_t} \\ &, \delta\mathbf{u_t})) + (A_t\delta\mathbf{x_t} + B_t\delta\mathbf{u_t} + r_t(\delta\mathbf{x_t}, \delta\mathbf{u_t}))^{\intercal} P_{t+1}(A_t\delta\mathbf{x_t} \\ &+ B_t\delta\mathbf{u_t} + r_t(\delta\mathbf{x_t}, \delta\mathbf{u_t})) + q_{t+1}(\delta\mathbf{x_{t+1}}).\end{aligned}$$

Similarly, expand the incremental cost at time t about the nominal state,

$$C_t(\mathbf{x_t}, \mathbf{u_t}) = \bar{l}_t + L_t\delta\mathbf{x_t} + \frac{1}{2}\delta\mathbf{x_t}^\intercal L_{tt}\delta\mathbf{x_t} + \frac{1}{2}\delta\mathbf{u_t}^\intercal R_t\bar{\mathbf{u}}_\mathbf{t} + \frac{1}{2}\bar{\mathbf{u}}_\mathbf{t}^\intercal R_t\delta\mathbf{u_t} + \frac{1}{2}\delta\mathbf{u_t}^\intercal R_t\delta\mathbf{u_t} + \frac{1}{2}\bar{\mathbf{u}}_\mathbf{t}^\intercal R_t\bar{\mathbf{u}}_\mathbf{t} + s_t(\delta\mathbf{x_t}).$$

$$\begin{aligned} J_t(\mathbf{x_t}, \mathbf{u_t}) &= \overbrace{[\bar{l}_t + \frac{1}{2}\bar{\mathbf{u}}_\mathbf{t}^\intercal R_t\bar{\mathbf{u}}_\mathbf{t} + \bar{J}_{t+1}(\bar{\mathbf{x}}_{\mathbf{t+1}})]}^{\bar{J}_t(\bar{\mathbf{x}}_\mathbf{t})} \\ &+ \delta\mathbf{u_t}^\intercal(B_t^\intercal\frac{P_{t+1}}{2}B_t + \frac{1}{2}R_t)\delta\mathbf{u_t} + \delta\mathbf{u_t}^\intercal(B_t^\intercal\frac{P_{t+1}}{2}A_t\delta\mathbf{x_t} \\ &+ \frac{1}{2}R_t\bar{\mathbf{u}}_\mathbf{t} + B_t^\intercal\frac{P_{t+1}}{2}r_t) + (\delta\mathbf{x_t}^\intercal A_t^\intercal\frac{P_{t+1}}{2}B_t + \frac{1}{2}\bar{\mathbf{u}}_\mathbf{t}R_t \\ &+ r_t^\intercal\frac{P_{t+1}}{2}B_t + G_{t+1}B_t)\delta\mathbf{u_t} + \delta\mathbf{x_t}^\intercal A_t^\intercal\frac{P_{t+1}}{2}A_t\delta\mathbf{x_t} \\ &+ \delta\mathbf{x_t}^\intercal\frac{P_{t+1}}{2}A_t^\intercal r_t + (r_t^\intercal\frac{P_{t+1}}{2}A_t + G_{t+1}A_t)\delta\mathbf{x_t} \\ &+ r_t^\intercal\frac{P_{t+1}}{2}r_t + G_{t+1}r_t + q_t. \end{aligned}$$

$$\text{Now, } \min_{\mathbf{u_t}} J_t(\mathbf{x_t}, \mathbf{u_t}) = \min_{\bar{\mathbf{u}}_\mathbf{t}} J_t(\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{u}}_\mathbf{t}) + \min_{\delta\mathbf{u_t}} H_t(\delta\mathbf{x_t}, \delta\mathbf{u_t})$$

First order optimality: At the optimal nominal control sequence $\bar{u}_t$, it follows from the minimum principle that

$$\begin{aligned} \frac{\partial C_t(\mathbf{x_t}, \mathbf{u_t})}{\partial \mathbf{u_t}} + \frac{\partial g(\mathbf{x_t})}{\partial \mathbf{u_t}}^\intercal\frac{\partial \bar{J}_{t+1}(\mathbf{x_{t+1}})}{\partial \mathbf{x_{t+1}}} &= 0 \\ \Rightarrow R_t\bar{\mathbf{u}}_\mathbf{t} + B_t^\intercal G_{t+1}^\intercal &= 0 \end{aligned} \tag{4}$$

By setting $\frac{\partial H_t(\delta\mathbf{x_t}, \delta\mathbf{u_t})}{\partial \delta\mathbf{u_t}} = 0$, we get:

$$\begin{aligned} \delta\mathbf{u_t}^* &= -S_t^{-1}(R_t\bar{\mathbf{u}}_\mathbf{t} + B_t^\intercal G_{t+1}^\intercal) - S_t^{-1}(B_t^\intercal P_{t+1}A_t + \\ &(G_t \otimes \tilde{R}_{t,xu})^\intercal)\delta\mathbf{x_t} - S_t^{-1}(B_t^\intercal P_{t+1}r_t) \\ &= \underbrace{-S_t^{-1}(B_t^\intercal P_{t+1}A_t + (G_{t+1} \otimes \tilde{R}_{t,xu})^\intercal)}_{K_t}\delta\mathbf{x_t} \\ &+ \underbrace{S_t^{-1}(-B_t^\intercal P_{t+1}r_t)}_{p_t} \end{aligned}$$

where, $S_t = R_t + B_t^\intercal P_{t+1}B_t$.

$$\Rightarrow \delta\mathbf{u_t} = K_t\delta\mathbf{x_t} + p_t.$$

Substituting it in the expansion of J_t and regrouping the terms based on the order of δx_t (till 2^{nd} order), we obtain:

$$\begin{aligned} \bar{J}_t(\mathbf{x_t}) &= \bar{J}_t(\bar{\mathbf{x}}_\mathbf{t}) + (L_t + (R_t\bar{\mathbf{u}}_\mathbf{t} + B_t^\intercal G_{t+1}^\intercal)K_t + G_{t+1}A_t)\delta\mathbf{x_t} \\ &+ \frac{1}{2}\delta\mathbf{x_t}^\intercal(L_{tt} + A_t^\intercal P_{t+1}A_t - K_t^\intercal S_t K_t + G_{t+1} \otimes \tilde{R}_{t,xx})\delta\mathbf{x_t}. \end{aligned}$$

Expanding the LHS about the optimal nominal state result in the equations (2) and (3).

REFERENCES

[1] R. Bellman, *Dynamic Programming*, 1^{st} edition, Princeton, NJ, USA: Princeton University Press, 1957.

[2] R. E. Kopp, "Pontryagin's maximum principle," *Mathematics in Science and Engineering*, vol. 5, pp. 255-279, 1962.

[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*. Proceedings of the 2005. IEEE, 2005, pp. 300 - 306.

[4] D. H. Jacobson and D. Q. Mayne, "Differential dynamic programming," 1970.

[5] E. Theodorou, Y. Tassa, and E. Todorov, "Stochastic differential dynamic programming," in *American Control Conference (ACC), 2010*. IEEE, 2010, pp. 1125 - 1132.

[6] P. Rutquist, "Methods for Stochastic Optimal Control," *PhD Thesis*, Chalmers University of Technology, pp. 9-12, 2017.

[7] D. Q. Mayne, "Model Predictive Control: Recent developments and future promise," *Automatica*, vol. 50, issue no. 12, pp. 2967-2986, Dec. 2014.

[8] N. Cazy, P.-B. Wieber, P. R. Giordano and F. Chaumette, "Visual Servoing Using Model Predictive Control to Assist Multiple Trajectory Tracking," *2017 IEEE International Conference on Robotics and Automation (ICRA 2017)*, 2017.

[9] G. Garimella, M. Sheckells and M. Kobilarov, "Robust Obstacle Avoidance for Aerial Platforms using Adaptive Model Predictive Control," *2017 IEEE International Conference on Robotics and Automation (ICRA 2017)*, 2017.

[10] G. Williams and N. Wagener and B. Goldfain and P. Drews and J. M. Rehg and B. Boots and E. A. Theodorou, Information theoretic MPC for model-based reinforcement learning, *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[11] E. Camacho and C. Bordons, "Nonlinear model predictive control: An introductory review," in *Assessment and Future Directions of Nonlinear Model Predictive Control*, ser. Lecture Notes in Control and Information Sciences. Springer Berlin Heidelberg, 2007, vol. 358, pp. 116.

[12] W.Langson, I.Chryssochoos and S.V. Raković and D.Q. Mayne, "Robust model predictive control using tubes", *Automatica*, vol. 40, pp.125-133, Elsevier, 2004.

[13] M. Rafieisakhaei, S. Chakravorty and P. R. Kumar, "A Near-Optimal Decoupling Principle for Nonlinear Stochastic Systems Arising in Robotic Path Planning and Control," *2017 IEEE* 56^{th} *Annual Conference on Decision and Control (CDC)*, 2017.

[14] M. Rafieisakhaei, S. Chakravorty and P. R. Kumar, "T-LQG : Closed-Loop Belief Space Planning via Trajectory-Optimized LQG," *2017 IEEE International Conference on Robotics and Automation (ICRA 2017)*, 2017.

[15] D. Bertsekas, Dynamic Programming and Optimal Control: 3rd Ed. Athena Scientific, 2007.

[16] N. Moshtagh, "Minimum volume enclosing ellipsoid, *Convex Optimization*, vol. 111, p. 112, 2005.

[17] V. Nevistic, J. Primbs, "Constrained nonlinear optimal control: A converse HJB approach," *Technical report 96-021*, California Institute of Technology, 1996.

[18] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, pp. 21492154, Sep. 2004.

[19] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, Andrew Ng, "ROS: an open-source Robot Operating System", *Open-source software workshop of the International Conference on Robotics and Automation*, Kobe, Japan, 2009.

[20] J.A.E. Andersson, J. Gillis, G. Horn, J.B. Rawlings and M. Diehl, "CasADi-A software framework for nonlinear optimization and optimal control", Mathematical Programming Computation, In press, 2018.

[21] A. Wächter and L. T. Biegler, "On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming", Mathematical Programming 106(1), pp. 25-57, 2006

[22] S.M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Technical Report 98-11, Iowa State University, October 1998.

[23] S.M. LaValle, Planning algorithms, *Cambridge University Press, 2006*.

[24] B. Kouvaritakis, M. Cannon (Eds.), "Nonlinear predictive control, theory and practice," London: The IEE, 2001.

[25] Y. Tassa, T. Erez, E. Todorov, "Synthesis and Stabilization of Complex Behaviors thorough Online Trajectory Optimization," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 49064913, Algarve, Portugal.

[26] W. H. Fleming, "Stochastic Control for Small Noise Intensities", SIAM J. Control, vol. 9, n. 3, pp. 473-517.

[27] T. Luukkonen, "Modelling and control of quadcopter, *Independent research project in applied mathematics*, Espoo, 2011.

[28] F.Furrer and M.Burri and M.Achtelik and R.Siegwart, "RotorS—A Modular Gazebo MAV Simulator Framework", Robot Operating System (ROS): The Complete Reference (Volume 1), 2016.