Continuous-Time Differential Dynamic Programming with Terminal Constraints

Wei Sun
wsun42@gatech.edu

Evangelos A. Theodorou
evangelos.theodorou@ae.gatech.edu

Panagiotis Tsiotras
tsiotras@gatech.edu

***Abstract*—In this work, we revisit the continuous-time Differential Dynamic Programming (DDP) approach for solving optimal control problems with terminal state constraints. We derive two algorithms, each for different order of expansion of the system dynamics and we investigate their performance in terms of their convergence speed. Compared to previous work, we provide a set of backward differential equations for the value function expansion by relaxing the assumption that the initial nominal control must be very close to the optimal control solution. We apply the derived algorithms to two classical optimal control problems, namely, the inverted pendulum and the Dreyfus rocket problem and show the benefit of second order expansion.**

I. INTRODUCTION

Differential Dynamic Programming (DDP) [1] is a well-known trajectory optimization method that iteratively finds a locally optimal control policy starting from a nominal control and state trajectory. Since its introduction in [1], there has been a plethora of variations and applications of DDP within the controls and robotics communities. Starting with a differential game theoretic formulation and its application to bipedal locomotion [2], [3] to receding horizon [4], and stochastic control formulations[5], [6], DDP has become one of the standard methods for trajectory optimization with a broad range of applications [5], [7], [8], [9], [10], [11], [12], [13], [14].

While DDP was initially derived for continuous-time problems, the bulk of the previous work on applications of DDP has focused on discrete time formulations of continuous-time optimal control problems. The key idea in the aforementioned discrete-time formulations is to first discretize the dynamics and then use Dynamic Programming (DP) to derive the backward propagation equations for the zeroth, first and second order approximation terms of the value function. Thus, instead of first optimizing to find the optimal control and then discretizing the solution so that it can be applied to a real physical system, in discrete time DDP discretization is performed first, followed by an optimization step to find the optimal control.

In this paper, our analysis focuses on continuous-time problems. In particular, and motivated by a restrictive assumption in the initial derivation of the continuous-time DDP in the book by D. H. Jacobson and D. Q. Mayne [1], we derive a new set of differential equations for the backward propagation of the value function for continuous-time optimal control problems with terminal state constraints. Specifically, the fundamental assumption in the derivation of continuous-time DDP in [1] is that the nominal control $\bar{\mathbf{u}}$ is close to the optimal control $\mathbf{u}^*$. This assumption allows the expansion of the terms in the Hamilton-Jacobi-Bellman (HJB) Partial Differential Equation (PDE) around $\mathbf{u}^*$ instead of $\bar{\mathbf{u}}$ and results in the cancelation of terms that depend on $\mathcal{H}_{\mathbf{u}^*} = 0$, where $\mathcal{H}_{\mathbf{u}^*}$ stands for the partial derivative of the Hamiltonian $\mathcal{H}$ with respect to the (optimal) control input.

The assumption of having a nominal control trajectory $\bar{\mathbf{u}}$ in the iterative process close to the optimal control $\mathbf{u}^*$ is restrictive and may increase the sensitivity of the overall optimization process with in terms of initialization and convergence. It is worth noting that the potential implications of this assumption were initially discussed in a review paper of the book on DDP [1] published in 1971 by Michael K. Sain [15]. However, to the best of our knowledge, there has not been an investigation of this issue, or its numerical implications, in relation to the order of the expansion of the dynamics and the form of the cost function under consideration. We believe that the main reason for this is the fact that most applications of DDP have been dominated by discrete-time formulations.

The proposed DDP derivation relaxes the assumption that $\bar{\mathbf{u}}$ must be very close to $\mathbf{u}^*$. Therefore, the quadratic expansions of the terms in the HJB PDE are computed around the nominal control $\bar{\mathbf{u}}$ and not the optimal control $\mathbf{u}^*$. In this case, the term $\mathcal{H}_{\bar{\mathbf{u}}}$ is not necessarily equal to zero. As we will see later on, the fact that $\mathcal{H}_{\bar{\mathbf{u}}} \neq 0$ has a number of important implications:

(a) The control update laws have both a feed-forward part and a feedback part, in contrast to the update law presented in [1] which has only feedback terms.
(b) The differential equations for the zeroth, first and second order expansions of the value function in our derivation also differ from those in [1]. Our approach allows one to choose an initial control that is not close to the optimal control, but still leads to convergence.
(c) While convergence of the proposed numerical scheme is similar for first and second order expansions of nonlinear dynamics affine in control, this is not always true when systems that are nonlinear in both state and control are considered. Our analysis identifies the cases where second order expansions of the dynamics are beneficial, at the expense, of course, of performing more computations per iteration, in order to compute the second order derivatives.

This work is organized as follows: in Section II we provide the problem formulation. Subsection II-A summarizes the different ways to expand the terms in the HJB PDE. In Section II-B we derive the set of differential equations for the backward propagation of the value function. In Section III we discuss the terminal conditions and the update of the Lagrange multiplier associated with the terminal constraint.

Finally, in Sections IV and V we provide simulation results for two example systems. We conclude the paper with some observations and some ideas for potential extensions.

II. Problem Formulation

In this section we derive the continuous-time DDP formulation with terminal state constraints. In particular, we consider the following optimal control problem with final constraint $\psi(\mathbf{x}(\mathbf{t_f}), \mathbf{t_f}) = 0$ and cost $J(\mathbf{x}, \mathbf{u})$,

$$\begin{aligned} &V(\mathbf{x}(t_0), \boldsymbol{\lambda}; t_0) = \min_{\mathbf{u}} J(\mathbf{x}, \mathbf{u}) \\ &= \min_{\mathbf{u}} \left[\phi(\mathbf{x}(\mathbf{t_f}), \mathbf{t_f}) + \boldsymbol{\lambda}^{\intercal} \psi(\mathbf{x}(\mathbf{t_f}), \mathbf{t_f}) + \int_{t_0}^{\mathbf{t_f}} \mathbf{L}(\mathbf{x}, \mathbf{u}, t) \mathrm{d}t \right], \end{aligned} \tag{1}$$

subject to the dynamics

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = F(\mathbf{x}, \mathbf{u}, t), \quad \mathbf{x}_0 = \mathbf{x}(t_0), \tag{2}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the state, $\mathbf{u} \in \mathbb{R}^m$ is the control and $\boldsymbol{\lambda} \in \mathbb{R}^p$ is the Lagrange multiplier. $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R} \to \mathbb{R}$ is the *running cost* and $(\phi(\mathbf{x}(\mathbf{t_f}), \mathbf{t_f}) + \boldsymbol{\lambda}^{\intercal} \psi(\mathbf{x}(\mathbf{t_f}), \mathbf{t_f}))$ is the *terminal cost*, where $\phi : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ and $\psi : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^p$. Given an initial or nominal trajectory of the state and control $(\bar{\mathbf{x}}, \bar{\mathbf{u}})$, and letting $\delta\mathbf{x} = \mathbf{x} - \bar{\mathbf{x}}$, $\delta\mathbf{u} = \mathbf{u} - \bar{\mathbf{u}}$, the linearized dynamics can be represented as

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = F(\bar{\mathbf{x}} + \delta\mathbf{x}, \bar{\mathbf{u}} + \delta\mathbf{u}, t), \tag{3}$$

$$\frac{\mathrm{d}\delta\mathbf{x}}{\mathrm{d}t} = F_{\mathbf{x}} \delta\mathbf{x} + F_{\mathbf{u}} \delta\mathbf{u} + \frac{\kappa}{2} \mathcal{F}, \tag{4}$$

where $\kappa = 1$ for 2nd order expansion of the dynamics and $\kappa = 0$ for 1st order expansion of dynamics, and $\mathcal{F} \in \mathbb{R}^n$. In (4) $\mathcal{F} = [\mathcal{F}^{(1)}, \dots, \mathcal{F}^{(n)}]^{\intercal}$, such that each element of $\mathcal{F}$ is given by

$$\mathcal{F}^{(i)} = \begin{bmatrix} \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix}^{\intercal} \begin{bmatrix} F^{(i)}_{\mathbf{xx}} & F^{(i)}_{\mathbf{xu}} \\ F^{(i)}_{\mathbf{ux}} & F^{(i)}_{\mathbf{uu}} \end{bmatrix} \begin{bmatrix} \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix}, \tag{5}$$

where $F = [F^{(1)}, \dots, F^{(n)}]^{\intercal}$ and where the arguments for $F_{\mathbf{x}}$, $F_{\mathbf{u}}$, $F^{(i)}_{\mathbf{xx}}$, $F^{(i)}_{\mathbf{xu}}$, etc are omitted for brevity. The parameter κ in (4) is introduced to differentiate the results derived from the first and second order expansion of the dynamics. A comparison between these two results will be discussed later on.

A. Expansions of the HJB equation

Our objective is to find the update law for the optimal control and the differential equations for the backward propagation of the Value function in (1). We start our analysis with the corresponding HJB equation

$$-\frac{\partial V(\mathbf{x}, \boldsymbol{\lambda}; t)}{\partial t} = \min_{\mathbf{u}} \left[\mathbf{L}(\mathbf{x}, \mathbf{u}, t) + V_{\mathbf{x}}(\mathbf{x}, \boldsymbol{\lambda}; t)^{\intercal} F(\mathbf{x}, \mathbf{u}, t) \right], \tag{6}$$

with boundary condition

$$V(\mathbf{x}, \boldsymbol{\lambda}; \mathbf{t_f}) = \phi(\mathbf{x}(\mathbf{t_f}), \mathbf{t_f}) + \boldsymbol{\lambda}^T \psi(\mathbf{x}(\mathbf{t_f}), \mathbf{t_f}). \tag{7}$$

By taking expansions of the term on the left-hand side of (6) around $(\bar{\mathbf{x}}, \bar{\mathbf{u}})$ up to second order, we obtain

$$\begin{aligned} &\frac{\partial V(\bar{\mathbf{x}} + \delta\mathbf{x}, \bar{\boldsymbol{\lambda}} + \delta\boldsymbol{\lambda}; t)}{\partial t} = \\ &\frac{\partial}{\partial t} \left(V(\bar{\mathbf{x}}, \bar{\boldsymbol{\lambda}}; t) + \begin{bmatrix} V_{\mathbf{x}} \\ V_{\boldsymbol{\lambda}} \end{bmatrix}^{\intercal} \begin{bmatrix} \delta\mathbf{x} \\ \delta\boldsymbol{\lambda} \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \delta\mathbf{x} \\ \delta\boldsymbol{\lambda} \end{bmatrix}^{\intercal} \begin{bmatrix} V_{\mathbf{xx}} & V_{\mathbf{x}\boldsymbol{\lambda}} \\ V_{\boldsymbol{\lambda}\mathbf{x}} & V_{\boldsymbol{\lambda}\boldsymbol{\lambda}} \end{bmatrix} \begin{bmatrix} \delta\mathbf{x} \\ \delta\boldsymbol{\lambda} \end{bmatrix} \right). \end{aligned} \tag{8}$$

For the value function $V(\mathbf{x}, \boldsymbol{\lambda}; t)$ and all its first and second order partial derivatives, we have

$$\frac{\partial}{\partial t}(\cdot) = \frac{\mathrm{d}}{\mathrm{d}t}(\cdot) - \frac{\partial}{\partial x}(\cdot)^{\intercal} F(\mathbf{x}, \mathbf{u}, t). \tag{9}$$

Therefore, the left-hand side of (6) becomes

$$\begin{aligned} &-\frac{\partial V(\mathbf{x}, \boldsymbol{\lambda}; t)}{\partial t} = \\ &-\frac{\mathrm{d}}{\mathrm{d}t} \left(V(\bar{\mathbf{x}}, \bar{\boldsymbol{\lambda}}; t) + \begin{bmatrix} V_{\mathbf{x}} \\ V_{\boldsymbol{\lambda}} \end{bmatrix}^{\intercal} \begin{bmatrix} \delta\mathbf{x} \\ \delta\boldsymbol{\lambda} \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \delta\mathbf{x} \\ \delta\boldsymbol{\lambda} \end{bmatrix}^{\intercal} \begin{bmatrix} V_{\mathbf{xx}} & V_{\mathbf{x}\boldsymbol{\lambda}} \\ V_{\boldsymbol{\lambda}\mathbf{x}} & V_{\boldsymbol{\lambda}\boldsymbol{\lambda}} \end{bmatrix} \begin{bmatrix} \delta\mathbf{x} \\ \delta\boldsymbol{\lambda} \end{bmatrix} \right) \\ &+ V_{\mathbf{x}}^{\intercal} F + \begin{bmatrix} \delta\mathbf{x} \\ \delta\boldsymbol{\lambda} \end{bmatrix}^{\intercal} \begin{bmatrix} V_{\mathbf{xx}} F \\ V_{\boldsymbol{\lambda}\mathbf{x}} F \end{bmatrix} + \frac{1}{2} \nabla_{\mathbf{x}} \left(\begin{bmatrix} \delta\mathbf{x} \\ \delta\boldsymbol{\lambda} \end{bmatrix}^{\intercal} \begin{bmatrix} V_{\mathbf{xx}} & V_{\mathbf{x}\boldsymbol{\lambda}} \\ V_{\boldsymbol{\lambda}\mathbf{x}} & V_{\boldsymbol{\lambda}\boldsymbol{\lambda}} \end{bmatrix} \begin{bmatrix} \delta\mathbf{x} \\ \delta\boldsymbol{\lambda} \end{bmatrix} \right) F. \end{aligned} \tag{10}$$

On the other hand,

$$V_{\mathbf{x}}(\bar{\mathbf{x}} + \delta\mathbf{x}, \bar{\boldsymbol{\lambda}} + \delta\boldsymbol{\lambda}; t) = V_{\mathbf{x}}(\bar{\mathbf{x}}, \bar{\boldsymbol{\lambda}}; t) + V_{\mathbf{xx}} \delta\mathbf{x} + V_{\mathbf{x}\boldsymbol{\lambda}} \delta\boldsymbol{\lambda} + \frac{1}{2} \mathcal{U},$$

where $\mathcal{U} \in \mathbb{R}^n$ and each element of $\mathcal{U}$ is defined as

$$\mathcal{U}^{(i)} = \begin{bmatrix} \delta\mathbf{x} \\ \delta\boldsymbol{\lambda} \end{bmatrix}^{\intercal} \begin{bmatrix} V^{(i)}_{\mathbf{xxx}} & V^{(i)}_{\mathbf{xx}\boldsymbol{\lambda}} \\ V^{(i)}_{\mathbf{x}\boldsymbol{\lambda}\mathbf{x}} & V^{(i)}_{\mathbf{x}\boldsymbol{\lambda}\boldsymbol{\lambda}} \end{bmatrix} \begin{bmatrix} \delta\mathbf{x} \\ \delta\boldsymbol{\lambda} \end{bmatrix},$$

where $V^{(i)}_{\mathbf{xxx}}$ denotes the Hessian matrix of the ith element of $V_{\mathbf{x}}$ and similarly for the others. Furthermore,

$$\begin{aligned} &\mathbf{L}(\bar{\mathbf{x}} + \delta\mathbf{x}, \bar{\mathbf{u}} + \delta\mathbf{u}, t) = \mathbf{L}(\bar{\mathbf{x}}, \bar{\mathbf{u}}, t) + \mathbf{L}_{\mathbf{x}}^{\intercal} \delta\mathbf{x} + \mathbf{L}_{\mathbf{u}}^{\intercal} \delta\mathbf{u} \\ &+ \frac{1}{2} \begin{bmatrix} \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix}^{\intercal} \begin{bmatrix} \mathbf{L}_{\mathbf{xx}} & \mathbf{L}_{\mathbf{xu}} \\ \mathbf{L}_{\mathbf{ux}} & \mathbf{L}_{\mathbf{uu}} \end{bmatrix} \begin{bmatrix} \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix}, \\ &F(\bar{\mathbf{x}} + \delta\mathbf{x}, \bar{\mathbf{u}} + \delta\mathbf{u}, t) = F(\bar{\mathbf{x}}, \bar{\mathbf{u}}, t) + F_{\mathbf{x}} \delta\mathbf{x} + F_{\mathbf{u}} \delta\mathbf{u} + \frac{\kappa}{2} \mathcal{F}. \end{aligned}$$

Thus, the right-hand side of (6) can be expressed as

$$\begin{aligned} &\min_{\mathbf{u}} \left[\mathbf{L}(\mathbf{x}, \mathbf{u}, t) + V_{\mathbf{x}}(\mathbf{x}, \boldsymbol{\lambda}; t)^{\intercal} F(\mathbf{x}, \mathbf{u}, t) \right] \\ &= \min_{\delta\mathbf{u}} \Bigg\{ \mathbf{L}(\bar{\mathbf{x}}, \bar{\mathbf{u}}, t) + \mathbf{L}_{\mathbf{x}}^{\intercal} \delta\mathbf{x} + \mathbf{L}_{\mathbf{u}}^{\intercal} \delta\mathbf{u} \\ &+ \frac{1}{2} \begin{bmatrix} \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix}^{\intercal} \begin{bmatrix} \mathbf{L}_{\mathbf{xx}} & \mathbf{L}_{\mathbf{xu}} \\ \mathbf{L}_{\mathbf{ux}} & \mathbf{L}_{\mathbf{uu}} \end{bmatrix} \begin{bmatrix} \delta\mathbf{x} \\ \delta\mathbf{u} \end{bmatrix} \\ &+ V_{\mathbf{x}}^{\intercal} F + V_{\mathbf{x}}^{\intercal} F_{\mathbf{x}}^{\intercal} \delta\mathbf{x} + V_{\mathbf{x}}^{\intercal} F_{\mathbf{u}}^{\intercal} \delta\mathbf{u} + \frac{\kappa}{2} V_{\mathbf{x}}^{\intercal} \mathcal{F} \\ &+ \delta\mathbf{x}^{\intercal} V_{\mathbf{xx}} F + \delta\mathbf{x}^{\intercal} V_{\mathbf{xx}} F_{\mathbf{x}}^{\intercal} \delta\mathbf{x} + \delta\mathbf{x}^{\intercal} V_{\mathbf{xx}} F_{\mathbf{u}}^{\intercal} \delta\mathbf{u} \\ &+ \delta\boldsymbol{\lambda}^{\intercal} V_{\boldsymbol{\lambda}\mathbf{x}} F + \delta\boldsymbol{\lambda}^{\intercal} V_{\boldsymbol{\lambda}\mathbf{x}} F_{\mathbf{x}}^{\intercal} \delta\mathbf{x} + \delta\boldsymbol{\lambda}^{\intercal} V_{\boldsymbol{\lambda}\mathbf{x}} F_{\mathbf{u}}^{\intercal} \delta\mathbf{u} \\ &+ \frac{1}{2} \begin{bmatrix} \delta\mathbf{x} \\ \delta\boldsymbol{\lambda} \end{bmatrix}^{\intercal} \begin{bmatrix} \sum\limits_{i=1}^{n} V^{(i)}_{\mathbf{xxx}} F^{(i)} & \sum\limits_{i=1}^{n} V^{(i)}_{\mathbf{x}\boldsymbol{\lambda}\mathbf{x}} F^{(i)} \\ \sum\limits_{i=1}^{n} V^{(i)}_{\boldsymbol{\lambda}\mathbf{xx}} F^{(i)} & \sum\limits_{i=1}^{n} V^{(i)}_{\boldsymbol{\lambda}\boldsymbol{\lambda}\mathbf{x}} F^{(i)} \end{bmatrix} \begin{bmatrix} \delta\mathbf{x} \\ \delta\boldsymbol{\lambda} \end{bmatrix} \\ &+ \text{h.o.t.} \Bigg\}. \end{aligned} \tag{11}$$

Note that

$$\nabla_{\mathbf{x}}\left(\begin{bmatrix}\delta\mathbf{x}\\ \delta\boldsymbol{\lambda}\end{bmatrix}^{\intercal}\begin{bmatrix}V_{\mathbf{xx}} & V_{\mathbf{x}\boldsymbol{\lambda}}\\ V_{\boldsymbol{\lambda}\mathbf{x}} & V_{\boldsymbol{\lambda}\boldsymbol{\lambda}}\end{bmatrix}\begin{bmatrix}\delta\mathbf{x}\\ \delta\boldsymbol{\lambda}\end{bmatrix}\right)F$$

$$= \sum_{i=1}^{n}\frac{\partial}{\partial\mathbf{x}^{(i)}}\left(\begin{bmatrix}\delta\mathbf{x}\\ \delta\boldsymbol{\lambda}\end{bmatrix}^{\intercal}\begin{bmatrix}V_{\mathbf{xx}} & V_{\mathbf{x}\boldsymbol{\lambda}}\\ V_{\boldsymbol{\lambda}\mathbf{x}} & V_{\boldsymbol{\lambda}\boldsymbol{\lambda}}\end{bmatrix}\begin{bmatrix}\delta\mathbf{x}\\ \delta\boldsymbol{\lambda}\end{bmatrix}\right)F^{(i)}$$

$$= \begin{bmatrix}\delta\mathbf{x}\\ \delta\boldsymbol{\lambda}\end{bmatrix}^{\intercal}\begin{bmatrix}\sum_{i=1}^{n}V_{\mathbf{xxx}}^{(i)}F^{(i)} & \sum_{i=1}^{n}V_{\mathbf{x}\boldsymbol{\lambda}\mathbf{x}}^{(i)}F^{(i)}\\ \sum_{i=1}^{n}V_{\boldsymbol{\lambda}\mathbf{xx}}^{(i)}F^{(i)} & \sum_{i=1}^{n}V_{\boldsymbol{\lambda}\boldsymbol{\lambda}\mathbf{x}}^{(i)}F^{(i)}\end{bmatrix}\begin{bmatrix}\delta\mathbf{x}\\ \delta\boldsymbol{\lambda}\end{bmatrix}.$$

After equating (10) with (11), and canceling repeated terms, we obtain

$$\begin{aligned}
&-\frac{\mathrm{d}}{\mathrm{d}t}\left(V(\bar{\mathbf{x}},\bar{\boldsymbol{\lambda}};t)+\begin{bmatrix}V_{\mathbf{x}}\\ V_{\boldsymbol{\lambda}}\end{bmatrix}^{\intercal}\begin{bmatrix}\delta\mathbf{x}\\ \delta\boldsymbol{\lambda}\end{bmatrix}+\frac{1}{2}\begin{bmatrix}\delta\mathbf{x}\\ \delta\boldsymbol{\lambda}\end{bmatrix}^{\intercal}\begin{bmatrix}V_{\mathbf{xx}} & V_{\mathbf{x}\boldsymbol{\lambda}}\\ V_{\boldsymbol{\lambda}\mathbf{x}} & V_{\boldsymbol{\lambda}\boldsymbol{\lambda}}\end{bmatrix}\begin{bmatrix}\delta\mathbf{x}\\ \delta\boldsymbol{\lambda}\end{bmatrix}\right)\\
&=\min_{\delta\mathbf{u}}\Bigg\{\mathbf{L}(\bar{\mathbf{x}},\bar{\mathbf{u}},t)+\mathbf{L}_{\mathbf{x}}^{\intercal}\delta\mathbf{x}+\mathbf{L}_{\mathbf{u}}^{\intercal}\delta\mathbf{u}\\
&+\frac{1}{2}\begin{bmatrix}\delta\mathbf{x}\\ \delta\mathbf{u}\end{bmatrix}^{\intercal}\begin{bmatrix}\mathbf{L}_{\mathbf{xx}} & \mathbf{L}_{\mathbf{xu}}\\ \mathbf{L}_{\mathbf{ux}} & \mathbf{L}_{\mathbf{uu}}\end{bmatrix}\begin{bmatrix}\delta\mathbf{x}\\ \delta\mathbf{u}\end{bmatrix}+V_{\mathbf{x}}^{\intercal}F_{\mathbf{x}}^{\intercal}\delta\mathbf{x}+V_{\mathbf{x}}^{\intercal}F_{\mathbf{u}}^{\intercal}\delta\mathbf{u}\\
&+\frac{\kappa}{2}\begin{bmatrix}\delta\mathbf{x}\\ \delta\mathbf{u}\end{bmatrix}^{T}\begin{bmatrix}\sum_{i=1}^{n}V_{\mathbf{x}}^{(i)}F_{\mathbf{xx}}^{(i)} & \sum_{i=1}^{n}V_{\mathbf{x}}^{(i)}F_{\mathbf{xu}}^{(i)}\\ \sum_{i=1}^{n}V_{\mathbf{x}}^{(i)}F_{\mathbf{ux}}^{(i)} & \sum_{i=1}^{n}V_{\mathbf{x}}^{(i)}F_{\mathbf{uu}}^{(i)}\end{bmatrix}\begin{bmatrix}\delta\mathbf{x}\\ \delta\mathbf{u}\end{bmatrix}\\
&+\delta\mathbf{x}^{\intercal}V_{\mathbf{xx}}F_{\mathbf{x}}^{\intercal}\delta\mathbf{x}\\
&+\delta\mathbf{x}^{\intercal}V_{\mathbf{xx}}F_{\mathbf{u}}^{\intercal}\delta\mathbf{u}+\delta\boldsymbol{\lambda}^{\intercal}V_{\boldsymbol{\lambda}\mathbf{x}}F_{\mathbf{x}}^{\intercal}\delta\mathbf{x}+\delta\boldsymbol{\lambda}^{\intercal}V_{\boldsymbol{\lambda}\mathbf{x}}F_{\mathbf{u}}^{\intercal}\delta\mathbf{u}\Bigg\}.
\end{aligned}\tag{12}$$

Note that in (10) with (11), the expansions are around the nominal control $\bar{\mathbf{u}}$. This is in contrast with the derivation of continuous-time DDP in [1] in which the expansion takes place around $(\mathbf{x}^*,\mathbf{u}^*)$. The key assumption in [1] is that $\bar{\mathbf{u}}$ is close to $\mathbf{u}^*$. To see the differences of the analysis, we define the Hamiltonian

$$\mathcal{H}=\mathbf{L}(\mathbf{x},\mathbf{u},t)+V_{\mathbf{x}}(\mathbf{x},\boldsymbol{\lambda};t)^{\intercal}F(\mathbf{x},\mathbf{u},t).\tag{13}$$

Expansion of the Hamiltonian will result in the term $\mathcal{H}_{\mathbf{u}^*}=\mathbf{L}_{\mathbf{u}}+F_{\mathbf{u}}^{\intercal}V_{\mathbf{x}}$. The fact that $\mathcal{H}_{\mathbf{u}^*}=0$ results in dropping terms from the derivation. Another implication of expanding the left-hand side of the HJB around $\mathbf{u}^*$ and the right-hand side of the same equations around $\bar{\mathbf{u}}$ is that there will be terms of the same order in $\delta\mathbf{x}$ on both sides of the HJB equation that will be unmatched and not easy to drop (see equations 2.2.19 and 2.2.20 in pages 20 and 21, as well as the analysis followed in pages 22 and 23 of [1]).

B. Optimal Control Variations

We now return back to our derivation, and specifically to equation (12). Our goal is to find the optimal control variation $\delta\mathbf{u}$. To find the $\delta\mathbf{u}$ that minimizes the last equation, we take the partial derivative of the right-hand side of (12) with respect to $\delta\mathbf{u}$ and set it equal to zero. After some calculations, we obtain

$$\begin{aligned}\delta\mathbf{u}&=-\mathbf{H}_{\mathbf{uu}}^{-1}(\mathbf{L}_{\mathbf{u}}+F_{\mathbf{u}}V_{\mathbf{x}})-\mathbf{H}_{\mathbf{uu}}^{-1}F_{\mathbf{u}}V_{\mathbf{x}\boldsymbol{\lambda}}\delta\boldsymbol{\lambda}\\ &-\mathbf{H}_{\mathbf{uu}}^{-1}\left(\frac{1}{2}\mathbf{H}_{\mathbf{ux}}+\frac{1}{2}\mathbf{H}_{\mathbf{xu}}^{\intercal}+F_{\mathbf{u}}V_{\mathbf{xx}}\right)\delta\mathbf{x},\end{aligned}\tag{14}$$

where the terms $\mathbf{H}_{\mathbf{uu}},\mathbf{H}_{\mathbf{ux}},\mathbf{H}_{\mathbf{xu}}$ are defined as

$$\mathbf{H}_{\mathbf{uu}}=\mathbf{L}_{\mathbf{uu}}+\kappa\sum_{i=1}^{n}V_{\mathbf{x}}^{(i)}F_{\mathbf{uu}}^{(i)},\tag{15}$$

$$\mathbf{H}_{\mathbf{ux}}=\mathbf{L}_{\mathbf{ux}}+\kappa\sum_{i=1}^{n}V_{\mathbf{x}}^{(i)}F_{\mathbf{ux}}^{(i)},\tag{16}$$

$$\mathbf{H}_{\mathbf{xu}}=\mathbf{L}_{\mathbf{xu}}+\kappa\sum_{i=1}^{n}V_{\mathbf{x}}^{(i)}F_{\mathbf{xu}}^{(i)}.\tag{17}$$

The optimal control variation can be rewritten as,

$$\delta\mathbf{u}=\mathbf{l}(t)+\mathbf{K}_{\mathbf{x}}(t)\delta\mathbf{x}+\mathbf{K}_{\boldsymbol{\lambda}}(t)\delta\boldsymbol{\lambda},\tag{18}$$

where the terms $\mathbf{l}(t),\mathbf{K}_{\mathbf{x}}(t)$ and $\mathbf{K}_{\boldsymbol{\lambda}}(t)$ are defined by

$$\begin{aligned}\mathbf{l}(t)&=-\mathbf{H}_{\mathbf{uu}}^{-1}(\mathbf{L}_{\mathbf{u}}+F_{\mathbf{u}}V_{\mathbf{x}}),\\ \mathbf{K}_{\mathbf{x}}(t)&=-\mathbf{H}_{\mathbf{uu}}^{-1}\left(\frac{1}{2}\mathbf{H}_{\mathbf{ux}}+\frac{1}{2}\mathbf{H}_{\mathbf{xu}}^{\intercal}+F_{\mathbf{u}}V_{\mathbf{xx}}\right),\\ \mathbf{K}_{\boldsymbol{\lambda}}(t)&=-\mathbf{H}_{\mathbf{uu}}^{-1}F_{\mathbf{u}}V_{\mathbf{x}\boldsymbol{\lambda}}.\end{aligned}\tag{19}$$

Substituting (18) into the HJB equation (12), we can derive the backward propagation equations with respect to the value function and all its first and second order partial derivatives. After some mathematical manipulations, (12) becomes

$$\begin{aligned}
&-\frac{\mathrm{d}}{\mathrm{d}t}\left(V+V_{\mathbf{x}}^{\intercal}\delta\mathbf{x}+V_{\boldsymbol{\lambda}}^{\intercal}\delta\boldsymbol{\lambda}+\frac{1}{2}\begin{bmatrix}\delta\mathbf{x}\\ \delta\boldsymbol{\lambda}\end{bmatrix}^{\intercal}\begin{bmatrix}V_{\mathbf{xx}} & V_{\mathbf{x}\boldsymbol{\lambda}}\\ V_{\boldsymbol{\lambda}\mathbf{x}} & V_{\boldsymbol{\lambda}\boldsymbol{\lambda}}\end{bmatrix}\begin{bmatrix}\delta\mathbf{x}\\ \delta\boldsymbol{\lambda}\end{bmatrix}\right)\\
&=\left(\mathbf{L}+\mathbf{L}_{\mathbf{u}}^{\intercal}\mathbf{l}+\frac{1}{2}\mathbf{l}^{\intercal}\mathbf{H}_{\mathbf{uu}}\mathbf{l}+V_{\mathbf{x}}^{\intercal}F_{\mathbf{u}}^{\intercal}\mathbf{l}\right)\\
&+\delta\mathbf{x}^{\intercal}\Big(\mathbf{L}_{\mathbf{x}}+F_{\mathbf{x}}V_{\mathbf{x}}+\mathbf{K}_{\mathbf{x}}^{\intercal}(\mathbf{L}_{\mathbf{u}}+F_{\mathbf{u}}V_{\mathbf{x}})+\frac{1}{2}\mathbf{H}_{\mathbf{xu}}\mathbf{l}\\
&\quad+\frac{1}{2}\mathbf{H}_{\mathbf{ux}}^{\intercal}\mathbf{l}+\mathbf{K}_{\mathbf{x}}^{\intercal}\mathbf{H}_{\mathbf{uu}}\mathbf{l}+V_{\mathbf{xx}}F_{\mathbf{u}}^{\intercal}\mathbf{l}\Big)\\
&+\delta\boldsymbol{\lambda}^{\intercal}\left(\mathbf{K}_{\boldsymbol{\lambda}}^{\intercal}\mathbf{L}_{\mathbf{u}}+\mathbf{K}_{\boldsymbol{\lambda}}^{\intercal}\mathbf{H}_{\mathbf{uu}}\mathbf{l}+V_{\boldsymbol{\lambda}\mathbf{x}}F_{\mathbf{u}}^{\intercal}\mathbf{l}\right)\\
&+\frac{1}{2}\begin{bmatrix}\delta\mathbf{x}\\ \delta\boldsymbol{\lambda}\end{bmatrix}^{\intercal}V_{\chi\chi}\begin{bmatrix}\delta\mathbf{x}\\ \delta\boldsymbol{\lambda}\end{bmatrix},
\end{aligned}\tag{20}$$

where $V_{\chi\chi}$ is a 2-by-2 block matrix, and

$$\begin{aligned}
V_{\chi\chi}(1,1)&=\mathbf{H}_{\mathbf{xx}}+\mathbf{K}_{\mathbf{x}}^{\intercal}\mathbf{H}_{\mathbf{uu}}\mathbf{K}_{\mathbf{x}}+\mathbf{H}_{\mathbf{xu}}^{\intercal}\mathbf{K}_{\mathbf{x}}+\mathbf{K}_{\mathbf{x}}^{\intercal}\mathbf{H}_{\mathbf{ux}}\\
&\quad+V_{\mathbf{xx}}F_{\mathbf{x}}^{\intercal}+F_{\mathbf{x}}V_{\mathbf{xx}}+V_{\mathbf{xx}}F_{\mathbf{u}}^{\intercal}\mathbf{K}_{\mathbf{x}}+\mathbf{K}_{\mathbf{x}}^{\intercal}F_{\mathbf{u}}V_{\mathbf{xx}},\\
V_{\chi\chi}(2,2)&=\mathbf{K}_{\boldsymbol{\lambda}}^{\intercal}\mathbf{H}_{\mathbf{uu}}\mathbf{K}_{\boldsymbol{\lambda}}+V_{\boldsymbol{\lambda}\mathbf{x}}F_{\mathbf{u}}^{\intercal}\mathbf{K}_{\boldsymbol{\lambda}}+\mathbf{K}_{\boldsymbol{\lambda}}^{\intercal}F_{\mathbf{u}}V_{\mathbf{x}\boldsymbol{\lambda}},\\
V_{\chi\chi}(1,2)&=\mathbf{H}_{\mathbf{xu}}\mathbf{K}_{\boldsymbol{\lambda}}+\mathbf{K}_{\mathbf{x}}^{\intercal}\mathbf{H}_{\mathbf{uu}}\mathbf{K}_{\boldsymbol{\lambda}}+F_{\mathbf{x}}V_{\mathbf{x}\boldsymbol{\lambda}}\\
&\quad+\mathbf{K}_{\mathbf{x}}^{\intercal}F_{\mathbf{u}}V_{\mathbf{x}\boldsymbol{\lambda}}+V_{\mathbf{xx}}F_{\mathbf{u}}^{\intercal}\mathbf{K}_{\boldsymbol{\lambda}},\\
V_{\chi\chi}(2,1)&=V_{\chi\chi}(1,2)^{\intercal},
\end{aligned}\tag{21}$$

where $\mathbf{H}_{\mathbf{xx}}:=\mathbf{L}_{\mathbf{xx}}+\kappa\sum_{i=1}^{n}V_{\mathbf{x}}^{(i)}F_{\mathbf{xx}}^{(i)}$.

The backward differential equations for the value function expansion terms can be specified by equating coefficients of the expressions on the left and right-hand side of (20). From (19), we have $\mathbf{L}_{\mathbf{u}}+F_{\mathbf{u}}V_{\mathbf{x}}=-\mathbf{H}_{\mathbf{uu}}\mathbf{l}$, $\frac{1}{2}\mathbf{H}_{\mathbf{ux}}+\frac{1}{2}\mathbf{H}_{\mathbf{xu}}^{\intercal}+F_{\mathbf{u}}V_{\mathbf{xx}}=-\mathbf{H}_{\mathbf{uu}}\mathbf{K}_{\mathbf{x}}$, and $F_{\mathbf{u}}V_{\mathbf{x}\boldsymbol{\lambda}}=-\mathbf{H}_{\mathbf{uu}}\mathbf{K}_{\boldsymbol{\lambda}}$. By taking these equations into account, the backward differential equations for the value function expansion terms can be simplified as

$$-\frac{\mathrm{d}V}{\mathrm{d}t}=\mathbf{L}-\frac{1}{2}\mathbf{l}^{\intercal}\mathbf{H}_{\mathbf{uu}}\mathbf{l},\tag{22}$$

$$-\frac{\mathrm{d}V_{\mathbf{x}}}{\mathrm{d}t} = \mathbf{L_x} + F_{\mathbf{x}}V_{\mathbf{x}} - \mathbf{K}_{\mathbf{x}}^{\intercal}\mathbf{H_{uu}l}, \tag{23}$$
$$-\frac{\mathrm{d}V_{\boldsymbol{\lambda}}}{\mathrm{d}t} = \mathbf{K}_{\boldsymbol{\lambda}}^{\intercal}\mathbf{L_u}, \tag{24}$$
$$-\frac{\mathrm{d}V_{\mathbf{xx}}}{\mathrm{d}t} = \mathbf{H_{xx}} - \mathbf{K}_{\mathbf{x}}^{\intercal}\mathbf{H_{uu}}\mathbf{K_x} + V_{\mathbf{xx}}F_{\mathbf{x}}^{\intercal} + F_{\mathbf{x}}V_{\mathbf{xx}}, \tag{25}$$
$$-\frac{\mathrm{d}V_{\boldsymbol{\lambda\lambda}}}{\mathrm{d}t} = -\mathbf{K}_{\boldsymbol{\lambda}}^{\intercal}\mathbf{H_{uu}}\mathbf{K}_{\boldsymbol{\lambda}}, \tag{26}$$
$$-\frac{\mathrm{d}V_{\mathbf{x}\boldsymbol{\lambda}}}{\mathrm{d}t} = \mathbf{H_{xu}}\mathbf{K}_{\boldsymbol{\lambda}} + F_{\mathbf{x}}V_{\mathbf{x}\boldsymbol{\lambda}} + V_{\mathbf{xx}}F_{\mathbf{u}}^{\intercal}\mathbf{K}_{\boldsymbol{\lambda}}. \tag{27}$$

III. Terminal Condition and Update of Lagrange Multiplier

In this section, we first specify the terminal condition for the backward differential equations with respect to the value function and all its first and second order partial derivatives. Then we give a brief derivation of the update law of $\delta\boldsymbol{\lambda}$. Finally, we put all the pieces together and present the algorithm of continuous-time DDP with terminal constraints.

At the final time $\mathbf{t_f}$ we have condition (7).

By taking the Taylor series expansions around $(\bar{\mathbf{x}}(\mathbf{t_f}), \bar{\boldsymbol{\lambda}})$ and equating coefficients, we obtain

$$\begin{aligned}
V(\mathbf{t_f}) &= \phi(\bar{\mathbf{x}}(\mathbf{t_f}), \mathbf{t_f}) + \bar{\boldsymbol{\lambda}}^{\intercal}\psi(\bar{\mathbf{x}}(\mathbf{t_f}), \mathbf{t_f}),\\
V_{\mathbf{x}}(\mathbf{t_f}) &= \phi_{\mathbf{x}}(\bar{\mathbf{x}}(\mathbf{t_f}), \mathbf{t_f}) + \psi_{\mathbf{x}}(\bar{\mathbf{x}}(\mathbf{t_f}), \mathbf{t_f})\bar{\boldsymbol{\lambda}},\\
V_{\boldsymbol{\lambda}}(\mathbf{t_f}) &= \psi(\bar{\mathbf{x}}(\mathbf{t_f}), \mathbf{t_f}),\\
V_{\mathbf{xx}}(\mathbf{t_f}) &= \phi_{\mathbf{xx}}(\bar{\mathbf{x}}(\mathbf{t_f}), \mathbf{t_f}) + \bar{\boldsymbol{\lambda}}^{\intercal}\psi_{\mathbf{xx}}(\bar{\mathbf{x}}(\mathbf{t_f}), \mathbf{t_f}),\\
V_{\boldsymbol{\lambda\lambda}}(\mathbf{t_f}) &= 0,\\
V_{\mathbf{x}\boldsymbol{\lambda}}(\mathbf{t_f}) &= \psi_{\mathbf{x}}(\bar{\mathbf{x}}(\mathbf{t_f})).
\end{aligned} \tag{28}$$

Let us now turn to the derivation of the update law of $\delta\boldsymbol{\lambda}$. We follow the procedure introduced in [1]. First, we expand the value function at $t = t_0$. Note that $x(t_0) = x_0$ is fixed.

$$V(x_0, \bar{\boldsymbol{\lambda}} + \delta\boldsymbol{\lambda}; t_0) = V(x_0, \bar{\boldsymbol{\lambda}}; t_0) + V_{\boldsymbol{\lambda}}^{\intercal}\delta\boldsymbol{\lambda} + \frac{1}{2}\delta\boldsymbol{\lambda}^{\intercal}V_{\boldsymbol{\lambda\lambda}}\delta\boldsymbol{\lambda}, \tag{29}$$

and $\boldsymbol{\lambda}$ should be chosen to minimize the value function, that is, $V_{\boldsymbol{\lambda}} + V_{\boldsymbol{\lambda\lambda}}\delta\boldsymbol{\lambda} = 0$, and thus $\delta\boldsymbol{\lambda} = -V_{\boldsymbol{\lambda\lambda}}^{-1}V_{\boldsymbol{\lambda}}|_{t=t_0}$. For numerical iterations and with further assumption that $\dot{V}_{\boldsymbol{\lambda}} \equiv 0$ [1], we set

$$\delta\boldsymbol{\lambda} = -\eta V_{\boldsymbol{\lambda\lambda}}^{-1}(t_0)V_{\boldsymbol{\lambda}}(\mathbf{t_f}), \tag{30}$$

where $\eta \in (0, 1]$ is a tuning parameter for controlling the step size of update of the Lagrange multiplier $\boldsymbol{\lambda}$.

The previous results are summarized in the DDP algorithm provided in Algorithm 1.

IV. Simulation Results

In this section, we will apply our algorithm with both first and second order expansion of the dynamics to two systems, namely, the inverted pendulum and the Dreyfus rocket problem. The dynamics of the first problem is affine in control and the cost is quadratic in control, whereas in the second problem, the dynamics are nonlinear in control and the cost function is non-quadratic. By applying the algorithm to these two systems, we want to demonstrate the efficiency of the DDP algorithm, and to make a comparison between the case of the first order and the second order expansion of the dynamics.

Algorithm 1 DDP Algorithm with Fixed Final Time

Input: Initial condition of the dynamics $\mathbf{x}_0$, initial control $\bar{\mathbf{u}}$, initial guess of the Lagrange multiplier $\bar{\boldsymbol{\lambda}}$, final time $\mathbf{t_f}$, tuning parameters γ and η. Set $\kappa = 0$ for the case of first order dynamics expansion, or $\kappa = 1$ for the case of second order dynamics expansion.

Output: Optimal Control $\mathbf{u}^*$ and the corresponding optimal gains $\mathbf{l}, \mathbf{K_x}, \mathbf{K}_{\boldsymbol{\lambda}}$.

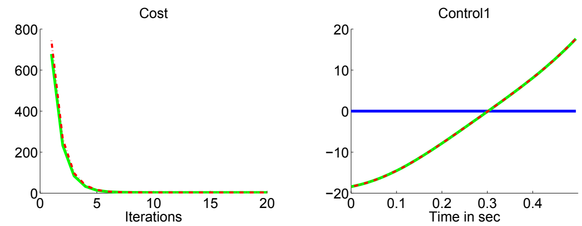
procedure UPDATE_CONTROL($\mathbf{x}_0$, $\bar{\mathbf{u}}$, $\bar{\boldsymbol{\lambda}}$, $\mathbf{t_f}$)
    while $\max(|\psi(\bar{\mathbf{x}}(\mathbf{t_f}), \mathbf{t_f})|) > \epsilon$, where $\max()$ returns the maximum element of a vector and $|\cdot|$ calculates the absolute value of each element in a vector, **do**
        Get the initial trajectory $\bar{\mathbf{x}}$ by integrating the dynamics forward with $\mathbf{x}_0$ and $\bar{\mathbf{u}}$;
        Compute the value of $V, V_{\mathbf{x}}, V_{\boldsymbol{\lambda}}, V_{\mathbf{xx}}, V_{\boldsymbol{\lambda\lambda}}, V_{\mathbf{x}\boldsymbol{\lambda}}$ at $\mathbf{t_f}$ according to (28);
        Integrate backward the Riccati equations (22) through (27) in which the partial derivatives with respect to $\mathbf{t_f}$ are all set to be 0;
        Compute $\delta\boldsymbol{\lambda}$ according to (30);
        Compute $\mathbf{l}, \mathbf{K_x}, \mathbf{K}_{\boldsymbol{\lambda}}$ from (19)
        Integrate (4) forward by replacing $\delta\mathbf{u}$ with $\mathbf{l} + \mathbf{K_x}\delta\mathbf{x} + \mathbf{K}_{\boldsymbol{\lambda}}\delta\boldsymbol{\lambda}$ to get $\delta\mathbf{x}$;
        Compute $\delta\mathbf{u} = \mathbf{l} + \mathbf{K_x}\delta\mathbf{x} + \mathbf{K}_{\boldsymbol{\lambda}}\delta\boldsymbol{\lambda}$;
        Update control $\mathbf{u}^* = \mathbf{u}^* + \gamma\delta\mathbf{u}$;
        Set $\bar{\mathbf{u}} = \mathbf{u}^*$ and $\bar{\boldsymbol{\lambda}} = \bar{\boldsymbol{\lambda}} + \delta\boldsymbol{\lambda}$, where $\delta\boldsymbol{\lambda} = -\eta V_{\boldsymbol{\lambda\lambda}}^{-1}(t_0)V_{\boldsymbol{\lambda}}(\mathbf{t_f})$;
    end while
    return $\mathbf{u}^*$, $\mathbf{l}$, $\mathbf{K_x}$, $\mathbf{K}_{\boldsymbol{\lambda}}$.
end procedure

A. Inverted Pendulum Problem

We first apply our algorithm on the inverted pendulum with soft terminal constraints, that is, the terminal constraint is of the form $\phi(\mathbf{x}(\mathbf{t_f}), \mathbf{t_f})$. In particular, the dynamics is given by $I\ddot{\theta} + b\dot{\theta} - mg\ell\sin\theta = \mathbf{u}$, where the parameters in the simulations are chosen as $m = 1$ Kg, $\ell = 0.5$ m, $b = 0.1$, $I = ml^2$, $g = 9.81\ \mathrm{Kg}\cdot\mathrm{m/sec}^2$. Our goal is to bring the pendulum from the initial state $[\theta, \dot{\theta}] = [\pi, 0]$ to $[\theta, \dot{\theta}] = [0, 0]$. The cost function is given by $J = \mathbf{x}(\mathbf{t_f})^{\intercal}Q_f\mathbf{x}(\mathbf{t_f}) + \int_0^{\mathbf{t_f}}\mathbf{u}^{\intercal}R\mathbf{u}$, where $\mathbf{x} = [\theta, \dot{\theta}]^{\intercal}$, $Q_f = \begin{bmatrix}100, & 0\\ 0, & 10\end{bmatrix}$ and $R = 0.1$.

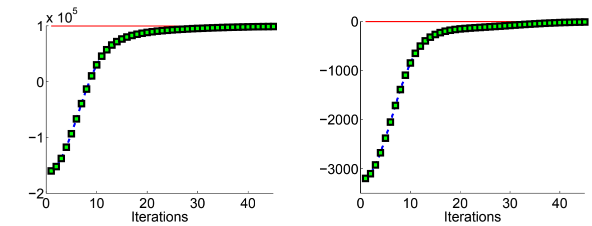
We set the initial control $\mathbf{u} \equiv 0$, the terminal time $\mathbf{t_f} = 0.5$, and the multiplier $\gamma = 0.4$. We run the algorithm for both cases $\kappa = 1$ and $\kappa = 0$. The results turn out to be quite similar. As can be seen in Figure 1a, in both cases the cost converges in six iterations. We include 20 iterations to ensure convergence. The optimal controls for the two cases at the 20th iteration are shown in Figure 1b, and they coincide with each other, as expected.

From this simulation, we see that when the problem is affine in control and the corresponding running cost is quadratic with respect to the control input, then the algorithm with the second order expansion of the dynamics may not provide much improvement compared to the algorithm with first order expansion of the dynamics. This behavior is probably expected.



(a) (b)

Fig. 1: (a) Cost per iteration when $\kappa = 0$ in dashed red, and cost when $\kappa = 1$ in green. (b) Optimal control u^* when $\kappa = 0$ in dashed red, u^* when $\kappa = 1$ in green, and initial control in blue.



(a) (b)

Fig. 2: First order expansion of dynamics, i.e. $\kappa = 0$, (a) Desired final altitude $x_2(\mathbf{t_f})$ in red, $x_2(\mathbf{t_f})$ per iteration in dashed blue. (b) Desired final vertical velocity component $x_4(\mathbf{t_f})$ in red, $x_4(\mathbf{t_f})$ per iteration in dashed blue.

B. Dreyfus Rocket Problem

Next, we apply our algorithm to the Dreyfus rocket problem [1]. In this problem, the task is to launch a rocket in fixed time such that to maximize the final horizontal velocity component, while specifying the final altitude and the final vertical velocity component. The dynamics of the rocket is given by

$$\begin{aligned} \dot{x}_1 &= x_3, \qquad \dot{x}_2 = x_4, \\ \dot{x}_3 &= a\cos u, \quad \dot{x}_4 = a\sin u - g, \end{aligned} \tag{31}$$

where x_1 is the horizontal distance, x_2 is the altitude, x_3 is the horizontal velocity component and x_4 is the vertical velocity component. The parameters are given by $a = 64, g = 32$. The initial condition is $[x_1(0), x_2(0), x_3(0), x_4(0)] = [0, 0, 0, 0]$, and the final time is fixed at $\mathbf{t_f} = 100$. The final constraints are

$$x_2(\mathbf{t_f}) - 100{,}000 = 0, \tag{32}$$

$$x_4(\mathbf{t_f}) = 0. \tag{33}$$

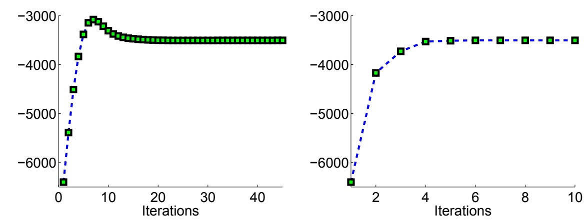
To maximize the final horizontal velocity component $x_3(\mathbf{t_f})$ is to minimize $-x_3(\mathbf{t_f}) = \int_0^{\mathbf{t_f}} (-a\cos u)\,\mathrm{d}t$. After adjoining the constraints, the cost function is given by

$$J(\mathbf{x}, u) = \boldsymbol{\lambda}^\intercal \psi(\mathbf{x}(\mathbf{t_f}), \mathbf{t_f}) + \int_0^{\mathbf{t_f}} (-a\cos u)\,\mathrm{d}t, \tag{34}$$

where $\boldsymbol{\lambda} = [\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2]^\intercal$ and $\psi(\mathbf{x}(\mathbf{t_f}), \mathbf{t_f}) = [x_2(\mathbf{t_f}) - 100{,}000, x_4(\mathbf{t_f})]^\intercal$. Before calculating the boundary conditions for the backward differential equations of the value function expansion, note that the relevant differential equations are

$$\dot{x}_2 = x_4, \quad \dot{x}_4 = a\sin u - g. \tag{35}$$

Hence, we only need to find the boundary conditions for the two states x_2 and x_4. From (28) we have $V(\mathbf{t_f}) = \boldsymbol{\lambda}_1(x_2(\mathbf{t_f}) - 100{,}000) + \boldsymbol{\lambda}_2 x_4(\mathbf{t_f}), V_{\mathbf{x}}(\mathbf{t_f}) = [\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2]^\intercal, V_{\boldsymbol{\lambda}}(\mathbf{t_f}) = [x_2(\mathbf{t_f}) - 100{,}000, x_4(\mathbf{t_f})]^\intercal, V_{\mathbf{xx}}(\mathbf{t_f}) = 0, V_{\boldsymbol{\lambda\lambda}}(\mathbf{t_f}) = 0, V_{\mathbf{x}\boldsymbol{\lambda}}(\mathbf{t_f}) = I$. First, we set $\kappa = 0$, which means that we consider the case of first order expansion of the dynamics. We start with the nominal control $\bar{\mathbf{u}} \equiv 0$ and the nominal Lagrange multiplier $\bar{\boldsymbol{\lambda}} = [0, 0]^\intercal$. There are also two other parameters that require tuning, γ, η, namely, the step sizes for the update of the control and the Lagrange multiplier, respectively. The largest value for γ that we could pick before the algorithm runs into error

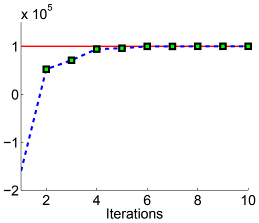


(a) (b)

Fig. 3: Dreyfus Rocket Problem: (a) Cost per iteration when $\kappa = 0$. (b) Cost per iteration when $\kappa = 1$.

is $\gamma = 0.1$. The reason why the error occurs is that in the algorithm, we need to take the inverse of $\mathbf{H_{uu}}$, which equals to $(a\cos u)$ when $\kappa = 0$. Hence, whenever the control u is close to $(k\pi + \pi/2), k \in \mathbb{Z}$, $\mathbf{H_{uu}}$ becomes very large, which, in turn, causes $\delta\mathbf{u}$ to be very large and leads to divergence and numerical instability. To limit the chance for the control to cross $(k\pi + \pi/2), k \in \mathbb{Z}$, we need to make γ small so that in every iteration, the update of control is not too far away from the previous control value. For the same reason, we set $\eta = 0.3$. The fixed final states are achieved in 45 iterations, as shown in Figure 2a and 2b. In Figure 3a, we can see that the cost becomes stable much earlier (after about 25 iterations) due to the existence of the Lagrange multiplier $\boldsymbol{\lambda}$. The optimal control is illustrated in Figure 5 with the dashed red line.

Next, we investigate the case when $\kappa = 1$, that is, we will use a second order expansion of the dynamics. Again, we start with the nominal control $\bar{\mathbf{u}} \equiv 0$ and nominal Lagrange multiplier $\bar{\boldsymbol{\lambda}} = [0, 0]^\intercal$. However, since $\mathbf{H_{uu}} = a\cos u + V_{\mathbf{x}}^\intercal [0, a\sin u]^\intercal$ when $\kappa = 1$, the possibility of the singularity in the previous case will not occur in this case. Moreover, we can set $\gamma = 1$ and $\eta = 1$. This will speed up the convergence significantly, and the optimal control is actually obtained in 6 iterations (10 iterations are included in the figures to show that the convergence is indeed achieved). The nominal and optimal controls are presented in Figure 5 in blue and green, respectively. The optimal controls in this case and in the previous one indeed coincide with each other. Figure 3b depicts the cost per iteration. Plots of the states $\mathbf{x}_2(\mathbf{t_f})$ and $\mathbf{x}_4(\mathbf{t_f})$ as a function of iteration number are shown in Figures 4a and



(a) (b)

Fig. 4: Second order expansion of dynamics, i.e. $\kappa = 1$, (a) Desired final altitude $x_2(\mathbf{t_f})$ in red, $x_2(\mathbf{t_f})$ per iteration in dashed blue. (b) Desired final vertical velocity component $x_4(\mathbf{t_f})$ in red, $x_4(\mathbf{t_f})$ per iteration in dashed blue.

4b, respectively.

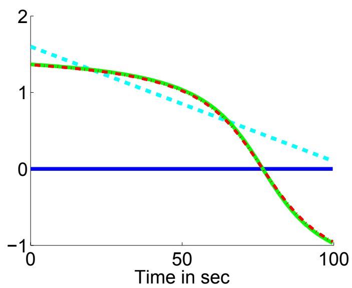


Fig. 5: Optimal control $\mathbf{u}^*$ when $\kappa = 0$ in dashed red, $\mathbf{u}^*$ when $\kappa = 1$ in green, our initial control in blue and initial control from [1] in dashed cyan.

This problem was also solved in [1]. The authors therein applied their version of DDP equations with the initial control $\bar{\mathbf{u}}(t) = 1.6 - 1.5t/100$, which is relatively close to the optimal control, as is shown in Figure 5. Their initial condition for the Lagrange multiplier $\bar{\boldsymbol{\lambda}} = [0.1, 1.0]^\intercal$ is also relatively close to the optimal value $\boldsymbol{\lambda}^* = [0.0632, 1.4900]^\intercal$.

By comparing these two cases, one sees that the second order expansion of the dynamics has a large impact on the convergence rate as well as the stability of the algorithm when the control is not affine in the dynamics.

V. Conclusion

In this paper, by dropping a restricted assumption in the previous derivation of the continuous-time DDP for optimal control problems with terminal state constraints, we find the update law and the backward propagation equations of the zeroth, first and second order approximation terms of the value function. One advantage of our approach lies in the fact that we do not need to assume that the initial nominal control and the optimal control solution are very close. Moreover, instead of using the discrete time formulation inherited by the previous work on applications of DDP, we present algorithms derived from first and second order expansion of the continuous dynamics. Specifically, we first find the optimal control in the continuous sense and then discretize the solution so that it can be applied to a real physical system.

We have tested the algorithms with first and second order expansion of dynamics on two distinct systems: one is the inverted pendulum where the control is affine in dynamics and quadratic in running cost, the other being the Dreyfus rocket where the control enters in trigonometric form in both the dynamics and running cost. In summary, when the control is affine in the dynamics and is quadratic in the running cost, the algorithm with first and second order expansion of the dynamics performs similarly. On the contrary, when the control is highly nonlinear in both the control and the dynamics, there is a performance improvement by applying the algorithm with the second order expansion of the dynamics compared to the one with the first order expansion of dynamics.

Future work includes application of the proposed algorithm into higher dimensional problems and address i) memory/time complexity requirements and ii) stopping criteria of the algorithm.

References

[1] D. H. Jacobson and D. Q. Mayne, *Differential dynamic programming*. New York,: American Elsevier Pub. Co., 1970.

[2] J. Morimoto, G. Zeglin, and C. Atkeson, "Minimax differential dynamic programming: application to a biped walking robot," in *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 2, Oct 2003, pp. 1927–1932 vol.2.

[3] J. Morimoto and C. Atkeson, "Minimax differential dynamic programming: An application to robust biped walking," in *In Advances in Neural Information Processing Systems 15*. Cambridge, MA: MIT Press, 2002.

[4] Y. Tassa, T. Erez, and W. D. Smart, "Receding horizon differential dynamic programming." in *NIPS*, J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, Eds. Curran Associates, Inc., 2007.

[5] 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*, June 2005, pp. 300–306 vol. 1.

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

[7] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *IROS*, 2012, pp. 4906–4913.

[8] ——, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, Oct 2012, pp. 4906–4913.

[9] P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng, "An application of reinforcement learning to aerobatic helicopter flight," in *In Advances in Neural Information Processing Systems 19*. MIT Press, 2007, p. 2007.

[10] T. Erez, Y. Tassa, and E. Todorov, "Infinite-horizon model predictive control for periodic tasks with contacts," in *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2011.

[11] Y. Tassa, N. Mansard, and E. Todorov, "Control-limited differential dynamic programming," in *Proceedings International Conference on Robotics and Automation*, 2014.

[12] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, Oct 2012, pp. 4906–4913.

[13] C. G. Atkeson and J. Morimoto, "Nonparametric representation of policies and value functions: A trajectory-based approach," in *Neural Information Processing Systems*. MIT Press, 2003, pp. 1611–1618.

[14] C. Atkeson and B. Stephens, "Random sampling of states in dynamic programming," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 38, no. 4, pp. 924–929, Aug 2008.

[15] W. Denham, "Differential dynamic programming," *Automatic Control, IEEE Transactions on*, vol. 16, no. 4, pp. 389–390, Aug 1971.