ISTS-2017-d-127／ISSFD-2017-127

Robust Differential Dynamic Programming for Low-Thrust Trajectory Design: Approach with Robust Model Predictive Control Technique

By Naoya OZAKI,[1)] Stefano CAMPAGNOLA,[2)] and Ryu FUNASE[1)]

[1)]*Department of Aeronautics and Astronautics, The University of Tokyo, Tokyo, Japan*
[2)]*Jet Propulsion Laboratory, NASA, Pasadena, California*

(Received June 21st, 2017)

Low-thrust propulsion is a key technology for space exploration, and much work in astrodynamics presents low-thrust trajectory design methods. Typically, a nominal trajectory is designed in a deterministic system. To account for model and execution errors, mission designers heuristically add margins - for example, by reducing the thrust and specific impulse or by computing penalties for specific failures. These conventional methods are time-consuming, done by hand by experts, and lead to conservative margins. This paper introduces a new method to compute nominal trajectories, taking into account disturbances. The proposed method, Tube Stochastic Differential Dynamic Programming, is a modified algorithm of Stochastic Differential Dynamic Programming to handle the control constraints. The proposed algorithm, which is inspired by the Tube MPC in the field of robotics, employs the sigma points to create a tube and computes the expected value by the Unscented Transform. Finally, we present numerical examples where the proposed solutions are more robust against disturbances when uncertainties are introduced.

Key Words: Low-Thrust Trajectory Optimization, Stochastic Control, DDP, Robust MPC

Nomenclature

$\boldsymbol{x}$	:	state vector: $\boldsymbol{x} \in \mathbb{R}^n$
$\boldsymbol{\mathcal{X}}$	:	sigma points on $\boldsymbol{x}$: $\boldsymbol{\mathcal{X}} \in \mathbb{R}^n$
$\boldsymbol{X}$	:	set of sigma points $\boldsymbol{\mathcal{X}}$: $\boldsymbol{X} \in \mathbb{R}^{2n(n+1)}$
$\boldsymbol{u}$	:	control vector: $\boldsymbol{u} \in \mathbb{U} \subset \mathbb{R}^m$
$\boldsymbol{\mathcal{U}}$	:	sigma points on $\boldsymbol{u}$: $\boldsymbol{\mathcal{U}} \in \mathbb{U} \subset \mathbb{R}^m$
$\boldsymbol{U}$	:	set of sigma points $\boldsymbol{\mathcal{U}}$: $\boldsymbol{U} \in \mathbb{R}^{2m(n+1)}$
$\boldsymbol{w}$	:	disturbance vector: $\boldsymbol{w} \in \mathbb{R}^n$
$\boldsymbol{\mathcal{W}}$	:	sigma points on $\boldsymbol{w}$: $\boldsymbol{\mathcal{W}} \in \mathbb{R}^n$
$\boldsymbol{\mu}$	:	mean value of random variable $\boldsymbol{x}$: $\boldsymbol{\mu} \in \mathbb{R}^n$
$\boldsymbol{P}$	:	covariance of random variable $\boldsymbol{x}$: $\boldsymbol{P} \in \mathbb{R}^{n\times n}$
$\boldsymbol{f}(\cdot)$	:	dynamical system: $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$
$\boldsymbol{h}(\cdot)$	:	control policy: $\boldsymbol{h} : \mathbb{R}^n \to \mathbb{R}^m$
$L(\cdot)$	:	cost functions: $L : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$
$\Phi(\cdot)$	:	terminal cost function: $\Phi : \mathbb{R}^n \to \mathbb{R}$
$V^*(\cdot)$	:	optimal cost-to-go function
$\boldsymbol{\alpha}$	:	open-loop control variations: $\boldsymbol{\alpha} \in \mathbb{R}^{2m(n+1)}$
$\boldsymbol{\beta}$	:	closed-loop control gains: $\boldsymbol{\beta} \in \mathbb{R}^{2m(n+1)\times 2n(n+1)}$
W_m	:	Weight on the sigma points for mean-value
W_c	:	Weight on the sigma points for covariance
n	:	dimension of state vectors
m	:	dimension of control vectors
Subscripts		
k	:	stage numbers: $k \in \{1, 2, ..., N+1\} \subset \mathbb{N}$
x	:	partial derivatives with respect to $\boldsymbol{x}$
u	:	partial derivatives with respect to $\boldsymbol{u}$
X	:	partial derivatives with respect to $\boldsymbol{X}$
U	:	partial derivatives with respect to $\boldsymbol{U}$

1. Introduction

Low-thrust propulsion is a key technology for space missions because of its high specific impulse, and various low-thrust trajectory design methods have been developed.[1,2)] One of the most numerically stable methods[3–7)] is Differential Dynamic Programming (DDP),[8)] which computes the optimal control solving a second order expansion of Bellman equation. Although low-thrust trajectory design methods are implemented on deterministic systems, in actual spacecraft operations, the trajectories are perturbed by disturbances including unmodeled accelerations, guidance/navigation errors, and missed-thrust[9)](i.e. the contingent coasting period due to operational troubles, such as safe-mode operations). For these reasons, mission designers use heuristic methods to add margins, for example introducing duty cycles,[9)] which reduce the thrust magnitude, and forced coast periods.[9)] These methods are time-consuming, tuned by hand by experts, and lead to conservative margins. As non-heuristic approaches, Olympio and Yam[10)] have suggested a surrogate-based method for one temporary engine failure, and Olympio[11)] has solved the same problem using two-stage stochastic programming. These methods include only one temporary engine failure and cannot model multiple engine failures or persistent disturbances. We should develop a systematic method to find the robust-optimal trajectory which guarantees the feasibility all along the trajectory when general uncertainties perturb the trajectory.

In the field of robotics, systematic methods have been developed to compute the optimal control in stochastic systems.[12–17)] One of the successful work is the constraint tightening model predictive control, which ensures the feasibility all along the trajectory when uncertainties perturb the nominal trajectory. This is achieved by tightening the constraints on the nominal control; in other words, it is accomplished by designing the nominal control without accelerating in full throttle and retaining the margin to compensate future perturbations. Another successful work is the tube model predictive control,[17)] which makes a tube around the nominal trajectory and ensures the feasibility to reach the target from the state inside the tube. These methods have been efficiently implemented in linear systems.

For nonlinear systems with uncertainties, Stochastic Differential Dynamic Programming (SDDP).[8, 18–20, 25] has been studied as the extension of DDP. However, these methods cannot handle the constraints, and therefore they are not practical for robust low-thrust trajectory design.

This paper presents a new SDDP-based algorithm to optimize the trajectory with uncertain dynamical systems and control constraints. The proposed method, Tube Stochastic Differential Dynamic Programming (TSDDP), is inspired by the tube model predictive control and the tube is created by the sigma points of the Unscented Transform.[21] Numerical examples show that our algorithm can be applied to low-thrust trajectory design problems, and the solution gains good robustness against disturbances without heuristic analyses.

2. Background of Stochastic Dynamic Programming

This section introduces the stochastic dynamical system and derives the stochastic dynamic programming with imperfect information. In other words, the optimization problems yield the optimal feedback control policies. In this work, 1) a stochastic dynamical system is modeled as the deterministic dynamical system with an additive disturbance, 2) the stochastic process is approximated as a Gaussian process, 3) the estimation errors are negligible and the control policy only depends on the estimated value.

2.1. Stochastic Dynamical System

This paper models the stochastic dynamical system of the spacecraft by a discrete-time stochastic equation with an additive disturbance

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}_k(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{w}_k, \quad k = 1, 2, ..., N \tag{1}$$

where $\boldsymbol{x}_k \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{P}_k)$ is a random state vector, $\boldsymbol{w}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_k)$ is a random disturbance vector, and $\boldsymbol{u}_k$ is the control vector specified by the control policy

$$\boldsymbol{u}_k = \boldsymbol{h}_k(\boldsymbol{x}_k) \tag{2}$$

Note that the stochastic process $\{\boldsymbol{x}_k\}$ does not keep a Gaussian process through the nonlinear transformation (1). Hereafter, we successively approximate $\{\boldsymbol{x}_k\}$ as a Gaussian process.

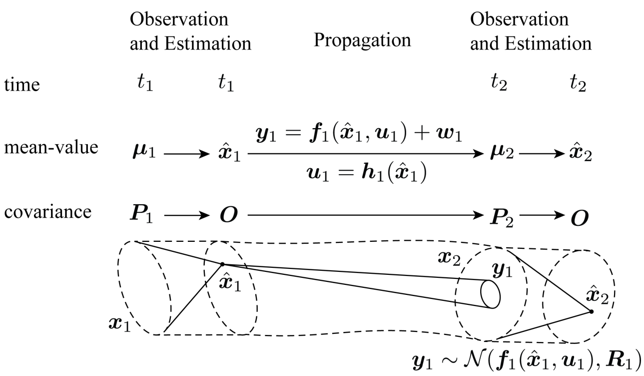


Fig. 1.: Example of stochastic process evolution with two stages.

Figure 1 illustrates the evolution of stochastic process $\{\boldsymbol{x}_k\}$. Let us explain the evolution from 1st stage to 2nd stage as an example. Starting from the initial condition $\boldsymbol{x}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1 \boldsymbol{P}_1)$ at the epoch t_1, the observation and estimation acquire the estimated state vector $\hat{\boldsymbol{x}}_1$. Here, we neglect the estimation errors for simplicity. The control vector $\boldsymbol{u}_1$ is determined through the control policy $\boldsymbol{h}_1(\cdot)$ with the estimated state vector $\hat{\boldsymbol{x}}_1$. The propagation with $(\hat{\boldsymbol{x}}_1, \boldsymbol{u}_1)$ yields the propagated state vector $\boldsymbol{y}_1$ by

$$\boldsymbol{y}_1 = \boldsymbol{f}_1(\hat{\boldsymbol{x}}_1, \boldsymbol{u}_1) + \boldsymbol{w}_1$$

where $\boldsymbol{y}_1 \sim \mathcal{N}(\boldsymbol{f}_1(\hat{\boldsymbol{x}}_1, \boldsymbol{u}_1), \boldsymbol{R}_1)$ because of the additive random disturbance $\boldsymbol{w}_1$. The trajectory design with a priori information must consider this process for all possible estimated state $\hat{\boldsymbol{x}}_1$. Therefore, the random state vector $\boldsymbol{x}_2$ at the epoch t_2 and the control vector $\boldsymbol{u}_1$ at the epoch t_1 are obtained as

$$\boldsymbol{x}_2 = \boldsymbol{f}_1(\boldsymbol{x}_1, \boldsymbol{u}_1) + \boldsymbol{w}_1$$
$$\boldsymbol{u}_1 = \boldsymbol{h}_1(\boldsymbol{x}_1)$$

where $\boldsymbol{x}_2$ should be approximated as a Gaussian random variable following $\mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{P}_2)$.

2.2. Stochastic Dynamic Programming

The distributions of Gaussian random variables are identified by the mean values and covariance matrices. Let us define the following mapping

$$b[\boldsymbol{x}_k] := \{(\boldsymbol{\mu}_k, \boldsymbol{P}_k) : \boldsymbol{x}_k \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{P}_k)\}, \tag{3}$$

and $b[\boldsymbol{x}_k]$ determines the property of the Gaussian random variable $\boldsymbol{x}_k$.

For a given mean value and covariance matrix $b[\boldsymbol{x}_k]$, the stochastic optimal control problem finds the optimal control policies $\boldsymbol{h}_{k:N}(\cdot)\,(=\{\boldsymbol{h}_k(\cdot), ..., \boldsymbol{h}_N(\cdot)\})$ to minimize the cost-to-go function

$$\begin{aligned} V_k(b[\boldsymbol{x}_k], \boldsymbol{h}_{k:N}(\cdot)) = \underset{\boldsymbol{x}_k}{E}\Big[L_k(\boldsymbol{x}_k, \boldsymbol{u}_k) + \underset{\boldsymbol{w}_k}{E}\,[L_{k+1}(\boldsymbol{x}_{k+1}, \boldsymbol{u}_{k+1}) + \cdots \\ + \underset{\boldsymbol{w}_{N-1}}{E}\Big[L_N(\boldsymbol{x}_N, \boldsymbol{u}_N) + \underset{\boldsymbol{w}_N}{E}\,[\Phi_{N+1}(\boldsymbol{x}_{N+1})|\boldsymbol{x}_N]\Big|\,\boldsymbol{x}_{N-1}\Big]\cdots\Big|\,\boldsymbol{x}_k\Big]\Big] \end{aligned} \tag{4}$$

where $\boldsymbol{x}_i, i = k+1, ..., N+1$ and $\boldsymbol{u}_i, i = k, ..., N$ are computed through Eqs.(1) and (2) with given $\boldsymbol{x}_k$. The optimal cost-to-go function is therefore given as

$$V_k^*(b[\boldsymbol{x}_k]) := \min_{\boldsymbol{h}_{k:N}(\cdot)} V_k(b[\boldsymbol{x}_k], \boldsymbol{h}_{k:N}(\cdot)) \tag{5}$$

Replacing the expected value $\underset{\boldsymbol{w}_k}{E}$ by $\underset{\boldsymbol{x}_{k+1}}{E}$ and introducing the Bellman's principle of optimality[27] derive a recursive equation

$$V_k^*(b[\boldsymbol{x}_k]) = \min_{\boldsymbol{h}_k(\cdot)}\left\{\underset{\boldsymbol{x}_k}{E}\left[L_k(\boldsymbol{x}_k, \boldsymbol{u}_k) + V_{k+1}^*(b[\boldsymbol{x}_{k+1}|\boldsymbol{x}_k])\right]\right\} \tag{6}$$

where

$$b[\boldsymbol{x}_{k+1}|\boldsymbol{x}_k] := \{(\boldsymbol{f}_k(\boldsymbol{x}_k, \boldsymbol{u}_k), \boldsymbol{R}_k) : \boldsymbol{x}_k \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{P}_k), \boldsymbol{u}_k = \boldsymbol{h}_k(\boldsymbol{x}_k)\}. \tag{7}$$

$b[\boldsymbol{x}_{k+1}|\boldsymbol{x}_k]$ computes the conditional expectation and its covariance where $\boldsymbol{x}_k$ is given. Note that $b[\boldsymbol{x}_{k+1}|\boldsymbol{x}_k]$ is also a random variable because $\boldsymbol{x}_k$ is a random variable.

Equation (6) is a typical formulation of Stochastic Dynamic Programming (SDP) with imperfect information.

3. Tube Stochastic Differential Dynamic Programming

This section proposes Tube Stochastic Differential Dynamic Programming (TSDDP) by introducing the Unscented Transform (UT). This algorithm, which is inspired by Tube MPC,[17] extends a conventional SDDP[8, 18–20, 25] to handle control constraints by making tubes of stochastic distribution.Introducing UT and the tubes of the sigma points can reformulate the stochastic dynamic programming as the deterministic dynamic programming. Once the problems are formulated as the deterministic dynamic programming, the recent techniques of DDP[5, 6, 28, 29] can efficiently solve the constrained problems.

3.1. Dynamical System with Unscented Transform

The Unscented Transform (UT) is a mathematical function to estimate the probability distribution as a Gaussian distribution. Given a nonlinear mapping $\boldsymbol{f}(\cdot)$ and an input random variable $\boldsymbol{x}$, the UT estimates the mean value and covariance of $\boldsymbol{f}(\boldsymbol{x})$. The UT refers the behaviors of the representative points, which is called *sigma points*, through the nonlinear transformation. The details are shown in Appendix A.

This section derives the deterministic dynamical system of sigma points instead of the stochastic dynamical system.[22, 23] Let us define the set of the sigma points $\boldsymbol{X}_k = [\boldsymbol{\mathcal{X}}_k^0, \boldsymbol{\mathcal{X}}_k^1, ..., \boldsymbol{\mathcal{X}}_k^{2n}] \in \mathbb{R}^{n(2n+1)}$. The sigma points of the random state vector are obtained from $b[\boldsymbol{x}_k]$ by

$$b[\boldsymbol{x}_k] \overset{\varphi_\sigma}{\longmapsto} \boldsymbol{X}_k \tag{8}$$

where the mapping φ_σ are defined in Appendix A

Instead of considering the control policies $\{\boldsymbol{h}_k(\cdot)\}$ as a function, let us express the control policies as the interpolant of the set of control vectors on the sigma points.

$$\boldsymbol{U}_k = [\boldsymbol{\mathcal{U}}_k^0, \boldsymbol{\mathcal{U}}_k^1, ...\boldsymbol{\mathcal{U}}_k^{2n}] \in \mathbb{R}^{m(2n+1)} \tag{9}$$

$$= [\boldsymbol{h}_k(\boldsymbol{\mathcal{X}}_k^0), \boldsymbol{h}_k(\boldsymbol{\mathcal{X}}_k^1), ...\boldsymbol{h}_k(\boldsymbol{\mathcal{X}}_k^{2n})] \tag{10}$$

3.1.1. Stochastic Dynamics with A Posteriori Information

Once the state vector is estimated as $\hat{\boldsymbol{x}}_k$, the control vector can be determined as $\boldsymbol{u}_k = \boldsymbol{h}_k(\hat{\boldsymbol{x}}_k)$. For the stochastic dynamics (1), the disturbance vector $\boldsymbol{w}_k$ perturbs the state vector at $(k+1)$-st stage. Introducing the sigma points with respect to $\boldsymbol{w}_k$ yields the sigma points of the propagated state vector $\boldsymbol{y}_k$ as

$$\boldsymbol{\mathcal{Y}}_k^p = \boldsymbol{f}_k(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k) + \boldsymbol{\mathcal{W}}_k^p, \quad p = 0, .., 2n \tag{11}$$

where $\boldsymbol{\mathcal{W}}_k^p \in \mathbb{R}^n, p = 1, .., 2n$ are the sigma point of $\boldsymbol{w}_k$.

3.1.2. Stochastic Dynamics with A Priori Information

Before the observation and estimation at the k-th stage, the state vector $\boldsymbol{x}_k$ cannot be determined as a fixed value. However, we can anticipate the distribution $b[\boldsymbol{x}_k]$ and it can be expressed by the set of sigma points $\boldsymbol{X}_k$. Therefore, the sigma points of the state vector $\boldsymbol{X}_{k+1}$ at $(k+1)$-st stage can be obtained from $\boldsymbol{X}_k$, $\boldsymbol{U}_k$ and $\boldsymbol{R}_k$ as

$$\boldsymbol{X}_{k+1} = \boldsymbol{F}_k(\boldsymbol{X}_k, \boldsymbol{U}_k, \boldsymbol{R}_k) \tag{12}$$

where $\boldsymbol{R}_k$ is the covariance matrix of $\boldsymbol{w}_k$. Equation (12) can be considered as the dynamical system with a priori information. The nonlinear mapping $\boldsymbol{F}_k : \mathbb{R}^{n(2n+1)} \times \mathbb{R}^{m(2n+1)} \times \mathbb{R}^{n \times n} \to \mathbb{R}^{n(2n+1)}$ can be derived in Appendix B

3.2. Unscented Stochastic Dynamical Programming

The Bellman equation (6) finds the optimal control policy $\boldsymbol{h}_k(\cdot)$ for given $b[\boldsymbol{x}_k]$ by introducing the optimal cost-to-go function $V_k^*(\cdot)$ with respect to $b[\boldsymbol{x}_k]$. The proposed algorithm replaces $b[\boldsymbol{x}_k]$ with the set of the sigma points $\boldsymbol{X}_k$, and finds the optimal control vectors $\boldsymbol{U}_k$ for the given $\boldsymbol{X}_k$.

Let us re-define the optimal cost-to-go function as $V_k^*(\boldsymbol{X}_k)$. The UT reformulates the Bellman equation (6) as a following recursive equation

$$V_k^*(\boldsymbol{X}_k) = \min_{\boldsymbol{U}_k} \sum_{j=0}^{2n} W_m^j \left[L_k(\boldsymbol{\mathcal{X}}_k^j, \boldsymbol{\mathcal{U}}_k^j) + V_{k+1}^*(\boldsymbol{Y}_k) \right] \tag{13}$$

where $V_{k+1}^*(\cdot)$ should be evaluated with a posteriori information, and therefore $\boldsymbol{Y}_k = [\boldsymbol{\mathcal{Y}}_k^0, \boldsymbol{\mathcal{Y}}_k^1, ..., \boldsymbol{\mathcal{Y}}_k^{2n}] \in \mathbb{R}^{n(2n+1)}$, and

$$\boldsymbol{\mathcal{Y}}_k^p = \boldsymbol{f}_k(\boldsymbol{\mathcal{X}}_k^j, \boldsymbol{\mathcal{U}}_k^j) + \boldsymbol{\mathcal{W}}_k^p, \quad p = 0, .., 2n. \tag{14}$$

Equation (13) is the formulation of the deterministic dynamic programming with respect to the sigma points $\boldsymbol{X}_k$.

We can realize that Eq.(13) has an analogy to Tube MPC.[17] Both the methods evaluate the cost-to-go function by taking the sum of cost $L_k(\cdot)$ at the representative points on the tubes. The proposed method adopts the sigma points as the representative points, and takes the weighted sum to evaluate the expected cost-to-go.

3.3. Tube Stochastic Differential Dynamic Programming

Tube Stochastic Differential Dynamic Programming (TSDDP) solves the Bellman equation (13) by Differential Dynamic Programming (DDP). The recent techniques of DDP[5, 6, 28, 29] can be adopted to solve the constrained problems efficiently because these methods can be applied to the deterministic dynamic programming such as Eq.(13).

3.3.1. Reference Trajectory

Let us introduce a reference trajectory $(\bar{\boldsymbol{X}}_k, \bar{\boldsymbol{U}}_k), k = 1, 2, ..., N$, where $\bar{\boldsymbol{X}}_k$ is the set of the sigma points of the reference trajectory and $\bar{\boldsymbol{U}}_k$ is the set of the control vectors on the sigma points. TSDDP can evaluate the control constraints only on the sigma points, and the constraints may not be satisfied outside sigma points region. Therefore, the sigma-points should be put on the 3-sigma ellipse for the practical use.

The UT has the arbitrary parameters $\alpha \in (0, 1]$ and $\kappa \in (0, \infty)$. The following condition is necessary and sufficient to put the sigma points on the 3-sigma ellipse

$$\kappa = \frac{9}{\alpha^2} - n \tag{15}$$

Proof. For the random variable $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{P})$, where $\boldsymbol{\mu} \in \mathbb{R}^n$ and $\boldsymbol{P} \in \mathbb{R}^{n \times n}$, the sigma points $\{\boldsymbol{\mathcal{X}}^0, \boldsymbol{\mathcal{X}}^1, ..., \boldsymbol{\mathcal{X}}^{2n}\}$ are calculated as

$$\boldsymbol{\mathcal{X}}^0 = \boldsymbol{\mu} \tag{16}$$

$$\boldsymbol{\mathcal{X}}^j = \boldsymbol{\mu} + \left(\sqrt{(n+\lambda)\boldsymbol{P}} \right)_j, \; j = 1, 2, ..., n \tag{17}$$

$$\boldsymbol{\mathcal{X}}^{n+j} = \boldsymbol{\mu} - \left(\sqrt{(n+\lambda)\boldsymbol{P}} \right)_j, \; j = 1, 2, ..., n \tag{18}$$

where $\lambda := \alpha^2(n+\kappa) - n$ and $(\cdot)_j$ means the j-th column vector of the matrix. To put the sigma points on the 3-sigma ellipse, the following condition should be satisfied

$$\sqrt{n+\lambda} = 3 \tag{19}$$

$$\Leftrightarrow \alpha^2(n+\kappa) = 9. \tag{20}$$

□

3.3.2. Second-Order Expansion of Bellman Equation

Let us expand Eq.(13) in the neighborhood of the reference trajectory $(\bar{\boldsymbol{X}}_k, \bar{\boldsymbol{U}}_k)$. The second order expansion of $L_k(\boldsymbol{\mathcal{X}}_k^j, \boldsymbol{\mathcal{U}}_k^j)$ is

$$L_k(\boldsymbol{\mathcal{X}}_k^j, \boldsymbol{\mathcal{U}}_k^j) \simeq L_{k,0} + \begin{bmatrix} \boldsymbol{L}_{k,x}^T & \boldsymbol{L}_{k,u}^T \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{\mathcal{X}}_k^j \\ \delta\boldsymbol{\mathcal{U}}_k^j \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \delta\boldsymbol{\mathcal{X}}_k^{jT} & \delta\boldsymbol{\mathcal{U}}_k^{jT} \end{bmatrix} \begin{bmatrix} \boldsymbol{L}_{k,xx} & \boldsymbol{L}_{k,xu} \\ \boldsymbol{L}_{k,xu}^T & \boldsymbol{L}_{k,uu} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{\mathcal{X}}_k^j \\ \delta\boldsymbol{\mathcal{U}}_k^j \end{bmatrix} \tag{21}$$

where $\delta\boldsymbol{\mathcal{X}}_k := \boldsymbol{\mathcal{X}}_k - \bar{\boldsymbol{\mathcal{X}}}_k$, $\delta\boldsymbol{\mathcal{U}}_k := \boldsymbol{\mathcal{U}}_k - \bar{\boldsymbol{\mathcal{U}}}_k$, and $L_{k,0}, \boldsymbol{L}_{k,x}, ..., \boldsymbol{L}_{k,uu}$ are evaluated at $(\bar{\boldsymbol{\mathcal{X}}}_k, \bar{\boldsymbol{\mathcal{U}}}_k)$.

$V_{k+1}^*(\cdot)$ can be expanded with respect to $(\boldsymbol{X}_k, \boldsymbol{U}_k)$ by introducing the chain rule. Let us first expand $V_{k+1}^*(\cdot)$ with respect to $\boldsymbol{Y}_k$ as follows

$$V_{k+1}^*(\boldsymbol{Y}_k) \simeq V_{k+1,0}^* + \boldsymbol{V}_{k+1,X}^{*T}\delta\boldsymbol{Y}_k + \frac{1}{2}\delta\boldsymbol{Y}_k^T \boldsymbol{V}_{k+1,XX}^*\delta\boldsymbol{Y}_k \tag{22}$$

where $\delta\boldsymbol{Y}_k = [\delta\boldsymbol{\mathcal{Y}}_k^0, \delta\boldsymbol{\mathcal{Y}}_k^1, ..., \delta\boldsymbol{\mathcal{Y}}_k^{2n}] \in \mathbb{R}^{n(2n+1)}$, and

$$\delta\boldsymbol{\mathcal{Y}}_k^p = \boldsymbol{\mathcal{Y}}_k^p - \bar{\boldsymbol{\mathcal{X}}}_{k+1}^p \tag{23}$$

$$= \boldsymbol{f}_k(\boldsymbol{\mathcal{X}}_k^j, \boldsymbol{\mathcal{U}}_k^j) + \boldsymbol{\mathcal{W}}_k^p - \bar{\boldsymbol{\mathcal{X}}}_{k+1}^p \tag{24}$$

Let $\boldsymbol{V}_{k+1,X}^* \in \mathbb{R}^{n(2n+1)}$ and $\boldsymbol{V}_{k+1,XX}^* \in \mathbb{R}^{n(2n+1)\times n(2n+1)}$ be partitioned into the block matrices with respect to sigma points as $\boldsymbol{\mathcal{V}}_{k+1,x}^p \in \mathbb{R}^n$ and $\boldsymbol{\mathcal{V}}_{k+1,xx}^{pq} \in \mathbb{R}^{n\times n}$. Introducing the block matrices simplifies Eq.(22) as

$$\begin{aligned} V_{k+1}^*(\boldsymbol{Y}_k) &\simeq V_{k+1,0}^* + \sum_{p=0}^{2n} \boldsymbol{\mathcal{V}}_{k+1,x}^{pT}\delta\boldsymbol{\mathcal{Y}}_k^p \\ &+ \frac{1}{2}\sum_{p=0}^{2n}\sum_{q=0}^{2n} \delta\boldsymbol{\mathcal{Y}}_k^{pT}\boldsymbol{\mathcal{V}}_{k+1,xx}^{pq}\delta\boldsymbol{\mathcal{Y}}_k^q \\ &= V_{k+1,0}^* + \sum_{p=0}^{2n} \boldsymbol{\mathcal{V}}_{k+1,x}^{pT}\left(\boldsymbol{\mathcal{W}}_k^p - \bar{\boldsymbol{\mathcal{X}}}_{k+1}^p\right) \\ &+ \frac{1}{2}\sum_{p=0}^{2n}\sum_{q=0}^{2n}\left(\boldsymbol{\mathcal{W}}_k^{pT} - \bar{\boldsymbol{\mathcal{X}}}_{k+1}^{pT}\right)\boldsymbol{\mathcal{V}}_{k+1,xx}^{pq}\left(\boldsymbol{\mathcal{W}}_k^q - \bar{\boldsymbol{\mathcal{X}}}_{k+1}^q\right) \\ &+ \sum_{p=0}^{2n}\sum_{q=0}^{2n}\left(\boldsymbol{\mathcal{W}}_k^{pT} - \bar{\boldsymbol{\mathcal{X}}}_{k+1}^{pT}\right)\boldsymbol{\mathcal{V}}_{k+1,xx}^{pq}\boldsymbol{f}_k \\ &+ \left(\sum_{p=0}^{2n}\boldsymbol{\mathcal{V}}_{k+1,x}^{pT}\right)\boldsymbol{f}_k + \frac{1}{2}\boldsymbol{f}_k^T\left(\sum_{p=0}^{2n}\sum_{q=0}^{2n}\boldsymbol{\mathcal{V}}_{k+1,xx}^{pq}\right)\boldsymbol{f}_k \end{aligned} \tag{25}$$

$$= A_{k+1} + \boldsymbol{B}_{k+1}^T\boldsymbol{f}_k + \frac{1}{2}\boldsymbol{f}_k^T\boldsymbol{C}_{k+1}\boldsymbol{f}_k \tag{26}$$

where $\boldsymbol{f}_k = \boldsymbol{f}_k(\boldsymbol{\mathcal{X}}_k^j, \boldsymbol{\mathcal{U}}_k^j)$ can be expanded as

$$\boldsymbol{f}_k(\boldsymbol{\mathcal{X}}_k^j, \boldsymbol{\mathcal{U}}_k^j) \simeq \boldsymbol{f}_{k,0} + \begin{bmatrix} \boldsymbol{f}_{k,x}^T & \boldsymbol{f}_{k,u}^T \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{\mathcal{X}}_k^j \\ \delta\boldsymbol{\mathcal{U}}_k^j \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \delta\boldsymbol{\mathcal{X}}_k^{jT} & \delta\boldsymbol{\mathcal{U}}_k^{jT} \end{bmatrix} \begin{bmatrix} \boldsymbol{f}_{k,xx} & \boldsymbol{f}_{k,xu} \\ \boldsymbol{f}_{k,xu}^T & \boldsymbol{f}_{k,uu} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{\mathcal{X}}_k^j \\ \delta\boldsymbol{\mathcal{U}}_k^j \end{bmatrix} \tag{27}$$

Finally, the Bellman Equation (13) can be expressed as the following quadratic form

$$\begin{aligned} V_k^*(\boldsymbol{X}_k) &= \min_{\delta\boldsymbol{U}_k}\sum_{j=0}^{2n} W_m^j\left\{q_0^j + \begin{bmatrix}\boldsymbol{q}_x^{jT} & \boldsymbol{q}_u^{jT}\end{bmatrix}\begin{bmatrix}\delta\boldsymbol{\mathcal{X}}_k^j \\ \delta\boldsymbol{\mathcal{U}}_k^j\end{bmatrix}\right. \\ &\left.+\frac{1}{2}\begin{bmatrix}\delta\boldsymbol{\mathcal{X}}_k^{jT} & \delta\boldsymbol{\mathcal{U}}_k^{jT}\end{bmatrix}\begin{bmatrix}\boldsymbol{q}_{xx}^j & \boldsymbol{q}_{xu}^j \\ \boldsymbol{q}_{xu}^{jT} & \boldsymbol{q}_{uu}^j\end{bmatrix}\begin{bmatrix}\delta\boldsymbol{\mathcal{X}}_k^j \\ \delta\boldsymbol{\mathcal{U}}_k^j\end{bmatrix}\right\} \end{aligned} \tag{28}$$

$$\begin{aligned} &= \min_{\delta\boldsymbol{U}_k}\left\{Q_0 + \begin{bmatrix}\boldsymbol{Q}_X^T & \boldsymbol{Q}_U^T\end{bmatrix}\begin{bmatrix}\delta\boldsymbol{X}_k \\ \delta\boldsymbol{U}_k\end{bmatrix}\right. \\ &\left.+\frac{1}{2}\begin{bmatrix}\delta\boldsymbol{X}_k^T & \delta\boldsymbol{U}_k^T\end{bmatrix}\begin{bmatrix}\boldsymbol{Q}_{XX} & \boldsymbol{Q}_{XU} \\ \boldsymbol{Q}_{XU}^T & \boldsymbol{Q}_{UU}\end{bmatrix}\begin{bmatrix}\delta\boldsymbol{X}_k \\ \delta\boldsymbol{U}_k\end{bmatrix}\right\} \end{aligned} \tag{29}$$

where the coefficients are defined in Appendix C. Eq.(29) can be solved by the ordinary DDP techniques.

3.3.3. Backward Sweep

The quadratic optimal control problem (29) yields the variation of the optimal control policy

$$\delta\boldsymbol{U}_k^* = \boldsymbol{\alpha}_k + \boldsymbol{\beta}_k\delta\boldsymbol{X}_k \tag{30}$$

where $\boldsymbol{\alpha}_k = -\boldsymbol{Q}_{UU}^{-1}\boldsymbol{Q}_U$ and $\boldsymbol{\beta}_k = -\boldsymbol{Q}_{UU}^{-1}\boldsymbol{Q}_{XU}$ if $\boldsymbol{Q}_{UU}$ is the positive definite matrix and the control constraints are neglected. If not, the special techniques are required. However, the optimal control policy can be still expressed as Eq.(30). Appendix D shows the strategy to solve the constrained DDP.

The left hand side of Eq.(29) is also expanded as

$$V_k^*(\boldsymbol{X}_k) \simeq V_{k,0}^* + \boldsymbol{V}_{k,x}^{*T}\delta\boldsymbol{X}_k + \frac{1}{2}\delta\boldsymbol{X}_k^T\boldsymbol{V}_{k,xx}^*\delta\boldsymbol{X}_k. \tag{31}$$

Substituting Eq.(30) to Eq.(29) and comparing the terms of the same order of $\delta\boldsymbol{X}_k$ yield the following equations

$$V_{k,0}^* = Q_0 + \boldsymbol{Q}_U^T\boldsymbol{\alpha}_k + \frac{1}{2}\boldsymbol{\alpha}_k^T Q_{uu}\boldsymbol{\alpha}_k \tag{32}$$

$$\boldsymbol{V}_{k,x}^{*T} = \boldsymbol{Q}_X^T + \boldsymbol{Q}_U^T\boldsymbol{\beta}_k + \boldsymbol{\alpha}_k^T\boldsymbol{Q}_{XU}^T + \boldsymbol{\alpha}_k^T\boldsymbol{Q}_{UU}\boldsymbol{\beta}_k \tag{33}$$

$$\boldsymbol{V}_{k,xx}^* = \boldsymbol{Q}_{XX} + \boldsymbol{Q}_{XU}\boldsymbol{\beta}_k + \boldsymbol{\beta}_k^T\boldsymbol{Q}_{XU}^T + \boldsymbol{\beta}_k^T\boldsymbol{Q}_{UU}\boldsymbol{\beta}_k \tag{34}$$

These quadratically expanded coefficients of $\boldsymbol{V}_k^*(\boldsymbol{X}_k)$ is recursively used to find the optimal control policies $\boldsymbol{\alpha}_{k-1}$ and $\boldsymbol{\beta}_{k-1}$ at $(k-1)$-st stage. Starting from the terminal cost function as the terminal condition

$$V_{N+1}^*(\boldsymbol{X}_{N+1}) = \sum_{i=0}^{2n} W_m^j\Phi_{N+1}(\boldsymbol{\mathcal{X}}_{N+1}^j), \tag{35}$$

the backward sweep process finds the optimal control policies $(\boldsymbol{\alpha}_k, \boldsymbol{\beta}_k)$, $k = 1, ..., N$ along the reference trajectory $(\bar{\boldsymbol{X}}_k, \bar{\boldsymbol{U}}_k)$.

3.3.4. Forward Sweep

The forward sweep process updates the reference trajectory $(\bar{\boldsymbol{X}}_k, \bar{\boldsymbol{U}}_k)$ by using the optimal control policies $(\boldsymbol{\alpha}_k, \boldsymbol{\beta}_k)$. The forward sweep process can simply propagate the trajectory because the optimal control policies have the feedback control terms $\boldsymbol{\beta}_k$ and Eq.(12) gives the dynamical system of the sigma points $\boldsymbol{X}_k$ is given as the deterministic system.

Table 1.: Configuration of Numerical Example.

Parameters	Settings
Time of Flight	$T_{ToF} = 756$ days
Segment	$N = 60$; i.e., $\Delta t = 12.6$ days
Gravity coef.	$GM = 1.327 \times 10^{11}$ km^3/s^2
Acceleration magn.	$u^{UB} = 4.15 \times 10^{-4}$ m/s^2
Initial condition	$[R_{x,1}, R_{y,1}] = [1.00, 0.00]$ a.u.
	$[V_{x,1}, V_{y,1}] = [0.00, 29.8]$ (km/s)
Final condition	$[R_{x,N+1}, R_{y,N+1}] = [2.24, -1.99]$ (km)
	$[V_{x,N+1}, V_{y,N+1}] = [11.4, 12.9]$ (km/s)
Standard deviation	$\sigma_r = 0.00$ (km)
	$\sigma_v = 9.42 \times 10^{-3}$ (km/s)
Weight	$c_u = 1.0$
	$c_1 = 1.0 \times 10^3, c_2 = 1.0 \times 10^3$
	$c_3 = 1.0 \times 10^3, c_4 = 1.0 \times 10^3$

4. Numerical Example

This section presents a numerical example to demonstrate that the Tube Stochastic Differential Dynamic Programming (TSDDP) yields robust low-thrust trajectory. The problem is to find the optimal control policies to minimize the expected delta-v and terminal distance from the target. Monte-Carlo simulation shows that SDDP trajectory obtains good robustness against disturbances, and the approximation by the Unscented Transform (UT) is close to the Monte Carlo results.

4.1. Statement of Problem

For the state vector $\boldsymbol{x} = \left[r_x, r_y, v_x, v_y\right]^T \in \mathbb{R}^4$ and the control vector $\boldsymbol{u} = [u_x, u_y]^T \in \mathbb{U} \subset \mathbb{R}^2$, where

$$\mathbb{U} := \left\{\boldsymbol{u} \in \mathbb{R}^2 : \|\boldsymbol{u}\| \leq u^{UB}\right\}, \tag{36}$$

the deterministic dynamical system is described as a planar two-body problem with respect to the Sun

$$\frac{d}{dt}\begin{bmatrix} r_x \\ r_y \\ v_x \\ v_y \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ -GM \cdot r_x/(r_x^2 + r_y^2)^{\frac{3}{2}} \\ -GM \cdot r_y/(r_x^2 + r_y^2)^{\frac{3}{2}} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ u_x \\ u_y \end{bmatrix} \tag{37}$$

where GM is the gravity constant of the Sun. Because TSDDP is formulated with discrete-time dynamical systems, Eq.(37) is discretized by the Runge-Kutta 4-th order method. Using the discretized dynamical system $\boldsymbol{f}_k(\cdot) : \mathbb{R}^4 \times \mathbb{R}^2 \to \mathbb{R}^4$, the discrete-time stochastic dynamical equation can be expressed as follows

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}_k(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{w}_k, k = 1, ..., N \tag{38}$$

where $\boldsymbol{x}_k, \boldsymbol{u}_k$ and $\boldsymbol{w}_k$ are the discretized state vector, control vector, and disturbance vector, respectively. The disturbance vector $\boldsymbol{w}_k$ is a random variable, whose covariance matrix is defined as

$$\boldsymbol{R}_k = \begin{bmatrix} \sigma_r^2 \cdot I_{2\times 2} & O_{2\times 2} \\ O_{2\times 2} & \sigma_v^2 \cdot I_{2\times 2} \end{bmatrix} \tag{39}$$

where $O_{2\times2} \in \mathbb{R}^{2\times2}$ is a null matrix, $I_{2\times2} \in \mathbb{R}^{2\times2}$ is an identity matrix, and the standard variation σ_r and σ_v use the number in Table 1.

The stochastic optimal control problem to minimize the expected delta-v is stated as follows. Given the initial condition and final condition, let us find the stochastic optimal control policy $\{\boldsymbol{U}_1^*, \boldsymbol{U}_2^*, ..., \boldsymbol{U}_N^*\}$ to minimize the expected-value of the cost-to-go function constructed by

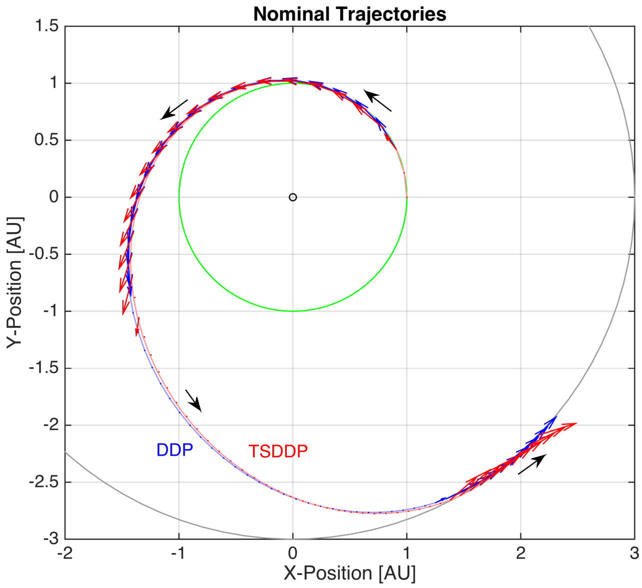


Fig. 2.: DDP and TSDDP nominal trajectories.

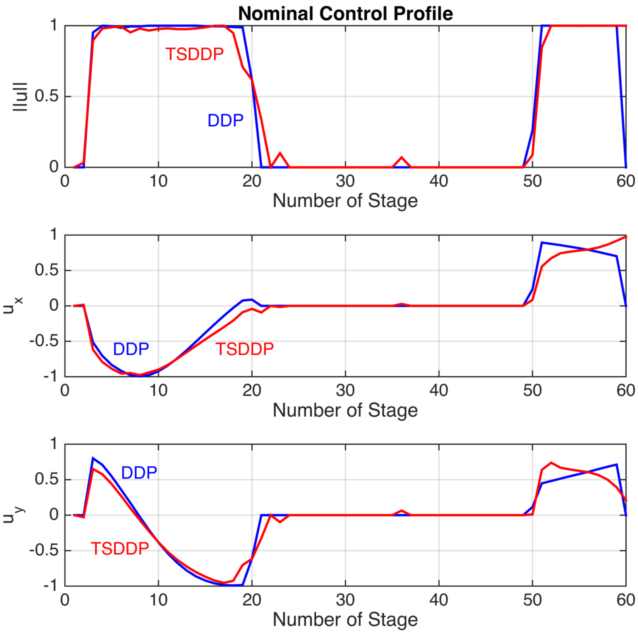


Fig. 3.: DDP and TSDDP nominal control profiles.

$$L_k(\boldsymbol{x}_k, \boldsymbol{u}_k) = c_u \|\boldsymbol{u}_k\| \delta t \tag{40}$$

$$\Phi_{N+1}(\boldsymbol{x}_{N+1}) = c_1(r_{x,N+1} - R_{x,N+1})^2 + c_2(r_{y,N+1} - R_{y,N+1})^2 + c_3(v_{x,N+1} - R_{x,N+1})^2 + c_4(v_{y,N+1} - R_{y,N+1})^2 \tag{41}$$

where the weights $c_u, c_1, ..., c_4$ are tuned as shown in Table 1. Note that the current version of TSDDP cannot include terminal constraints, which are difficult to impose stably because the boundary constraint is violated by the random perturbation. Future work will extend TSDDP to handle the terminal boundary condition by using, for example, the disturbance invariant set in the field of Robust Model Predictive Control.[17]

4.2. Numerical Results of Nominal Trajectories

We have solved both the non-disturbed optimal control problem by DDP and disturbed stochastic optimal control problem by TSDDP in order to show the difference. Figures 2 and 3 illustrate both the nominal trajectories and control profiles. The nominal control profile of TSDDP is different from one of DDP, and this difference contributes to improving the robustness against uncertainties in TSDDP. The control norm of TSDDP represents that the first thrusting arc is not actively constrained. This result indicates that TSDDP achieves the solution of the constraint tightening model predictive control,[12] and it retains a "margin" in the optimization for future feedback control. The future work should make the control profile smooth by introducing analytical derivatives of state transition matrix[30] and the conjugate Unscented Transform.[31]

4.3. Optimal Control Policy

TSDDP renders the optimal control vectors $\boldsymbol{U}_k$ at the sigma points, and therefore the control policy $\boldsymbol{h}_k(\cdot)$ should be created by interpolating these control vectors $\boldsymbol{U}_k$. Let us adopt the linear interpolant to create the control policy.[17]

For $\boldsymbol{U}_k = \{\mathcal{U}_k^0, \mathcal{U}_k^1, ..., \mathcal{U}_k^{2n}\}$, where $\mathcal{U}_k^j$ is associated with the sigma points $\boldsymbol{X}_k^j$, the control policy is defined as

$$\boldsymbol{h}_k(\boldsymbol{x}) = \sum_{j=0}^{2n} \lambda^j(\boldsymbol{x})\mathcal{U}_k^j \tag{42}$$

where $\boldsymbol{\lambda}(\boldsymbol{x}) = [\lambda^0(\boldsymbol{x}), \lambda^1(\boldsymbol{x}), ..., \lambda^{2n}(\boldsymbol{x})]^T$ is a least square solution of

$$\sum_{j=0}^{2n} \lambda^j(\boldsymbol{x})\boldsymbol{X}_k^j = \boldsymbol{x} \tag{43}$$

subject to

$$\sum_{j=0}^{2n} \lambda^j = 1. \tag{44}$$

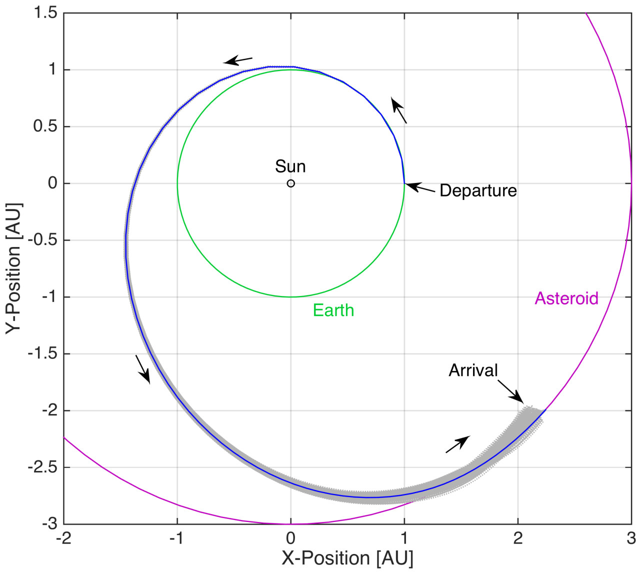


Fig. 4.: DDP Trajectories.

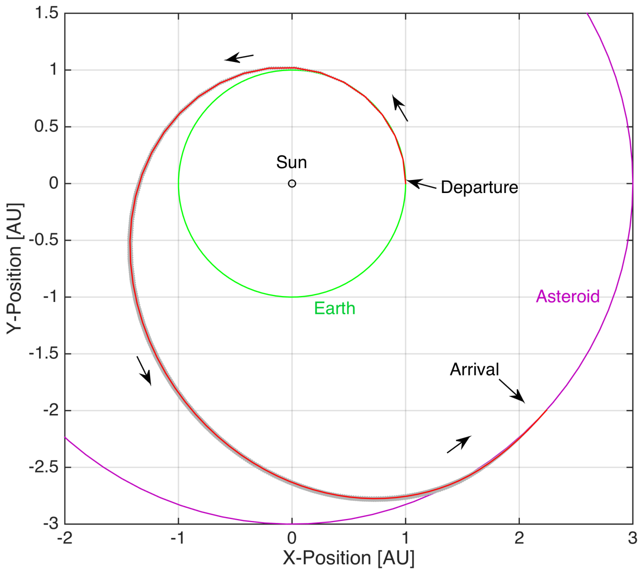


Fig. 6.: TSDDP Trajectories (Monte-Carlo).

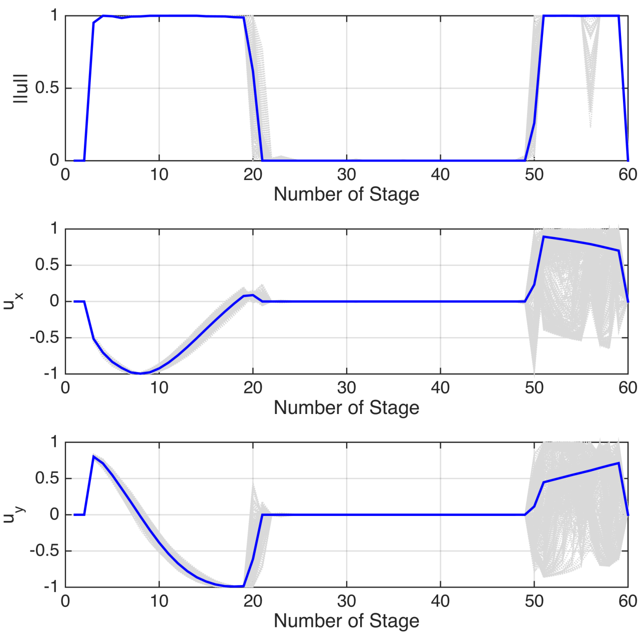


Fig. 5.: DDP control profiles.

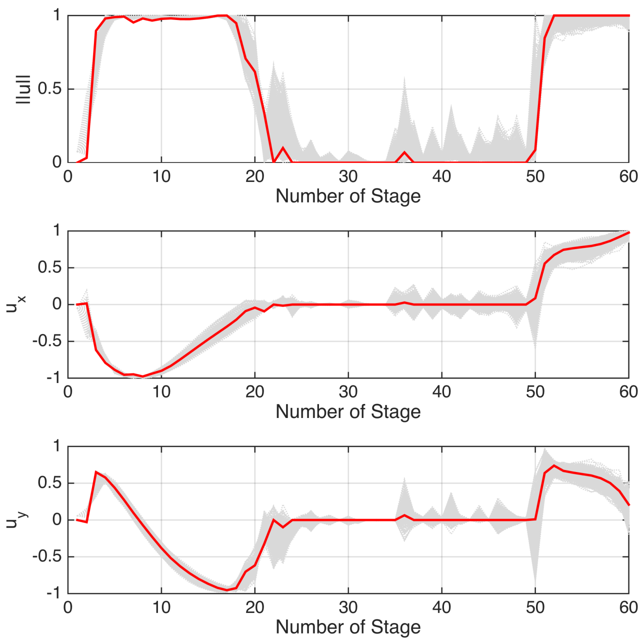


Fig. 7.: TSDDP control profiles.

4.4. Evaluation by Monte-Carlo Method

This section demonstrates the robustness of TSDDP solution by the Monte-Carlo method. The Monte-Carlo simulation randomly adds the disturbance in the dynamical system, and the optimal control policy $\{\boldsymbol{h}_k\}$ gives feedback to the perturbed trajectory. The details of the Monte-Carlo method are described below.

For every k-th stage

1. Given the state vector $\boldsymbol{x}_k$ as a deterministic value, compute $\lambda(\boldsymbol{x}_k)$ by Eqs. (43) and (44)
2. Compute the control vector $\boldsymbol{u}_k$ through the control policy $\boldsymbol{h}_k(\boldsymbol{x}_k)$ in Eq.(42).
3. Select a sample of the random disturbance $\boldsymbol{w}_k$, whose covariance is defined in Eq.(39), and compute $\boldsymbol{x}_{k+1}$ with given $\boldsymbol{x}_k, \boldsymbol{u}_k$, and $\boldsymbol{w}_k$ through Eq.(38).

DDP also produces the optimal control policies $\boldsymbol{h}_k(\boldsymbol{x}_k)$ including the feedback against perturbations. For the Monte-Carlo simulation with the DDP trajectory, the optimal control policies $\boldsymbol{h}_k(\boldsymbol{x}_k)$ in the procedure 2 should be replaced as

$$\boldsymbol{u}_k = \boldsymbol{h}_k(\boldsymbol{x}_k) = \bar{\boldsymbol{u}}_k + \boldsymbol{\beta}_k(\boldsymbol{x}_k - \bar{\boldsymbol{x}}_k) \tag{45}$$

where $\bar{\boldsymbol{u}}_k$ and $\bar{\boldsymbol{x}}_k$ are the nominal control and state vector, respectively.

Figures 4 and 6 plot 500 samples of the trajectories obtained by the Monte-Carlo simulation. As drawn in Fig. 4, the most of the DDP trajectories cannot cancel the perturbation and do not satisfy the terminal boundary condition, while most of the TSDDP trajectories cancel the perturbations, as shown in Fig. 6. The control profiles are illustrated in Figs. 5 and 7. The control profiles of DDP are diverged around the final phase because the perturbations are too large to compensate, but those of the TSDDP can correct the perturbations by adjusting the thrust vectors.

Figure 8 shows the evolution of the stochastic process. The distributions of the Monte-Carlo samples coincide with the error ellipse of the Unscented Transform very well. However, the sample points differ from the error ellipse, for example, at the stage $k = 47$ and $k = 60$. This difference comes from the nonlinearity of the dynamical system. Future work should introduce the conjugate Unscented Transform[31)] to evaluate the stochastic distribution accurately.

5. Conclusion

This paper proposes a new method to compute optimal low-thrust trajectories taking into account disturbances. The method is Tube Stochastic Differential Dynamic Programming (TSDDP), which is based on Stochastic Differential Dynamic Programming (SDDP) and inspired by the tube model predictive control. The proposed algorithm can solve the stochastic optimal control problem with control constraints by the tube created by the sigma points, and TSDDP can efficiently compute the stochastic optimal control problem by the Unscented Transform (UT). The Monte-Carlo simulation shows that the SDDP trajectories have more robustness to disturbances than the DDP trajectories, and the approximation by UT is accurate enough to achieve the robust low-thrust trajectory.

Acknowledgments

This work has been supported by JSPS Grant-in-Aid for JSPS Fellows Number 15J05999.

AppendixA Unscented Transform

The Unscented Transform (UT) is a mathematical function to estimate the probability distribution as a Gaussian distribution. The UT refers the behaviors of the representative points, which is called *sigma points*, through the nonlinear transformation.Let us introduce the random variable $\boldsymbol{x} \in \mathbb{R}^n$ and the nonlinear transformation $\boldsymbol{f}(\cdot) : \mathbb{R}^n \to \mathbb{R}^m$. For given $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$ and $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_x, \boldsymbol{P}_x)$, the UT estimate the mean value and covariance of $\boldsymbol{y}$, i.e. $\boldsymbol{\mu}_y$ and $\boldsymbol{P}_y$ where $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}_y, \boldsymbol{P}_y)$.

Let us define the set of the sigma points with respect to $\boldsymbol{x}$ as $\boldsymbol{X} = [\boldsymbol{\mathcal{X}}^0, \boldsymbol{\mathcal{X}}^1, ..., \boldsymbol{\mathcal{X}}^{2n}] \in \mathbb{R}^{n(2n+1)}$, where

$$\boldsymbol{\mathcal{X}}^0 = \boldsymbol{\mu}_x, \tag{46}$$

$$\boldsymbol{\mathcal{X}}^j = \boldsymbol{\mu}_x + \left(\sqrt{(n+\lambda)\boldsymbol{P}_x}\right)_j, j = 1, 2, ..., n, \tag{47}$$

$$\boldsymbol{\mathcal{X}}^{n+j} = \boldsymbol{\mu}_x - \left(\sqrt{(n+\lambda)\boldsymbol{P}_x}\right)_j, j = 1, 2, ..., n, \tag{48}$$

$\lambda := \alpha^2(n+\kappa) - n$, and $(\cdot)_j$ represents the j-th column vector of the matrix.

The transformed sigma-points is therefore

$$\boldsymbol{\mathcal{Y}}^j = \boldsymbol{f}(\boldsymbol{\mathcal{X}}^j), j = 0, ..., 2n \tag{49}$$

and the set of the transformed points are $\boldsymbol{Y} =$

The UT derives the approximated mean value and covariance by taking the weighted sum of the transformed sigma points as

$$\boldsymbol{\mu}_y = \sum_{j=0}^{2n} W_m^j \boldsymbol{\mathcal{Y}}^j \tag{50}$$

$$\boldsymbol{P}_y = \sum_{j=0}^{2n} W_c^j \left\{\boldsymbol{\mathcal{Y}}^j - \boldsymbol{\mu}_y\right\}\left\{\boldsymbol{\mathcal{Y}}^j - \boldsymbol{\mu}_y\right\}^T \tag{51}$$

where

$$W_m^0 = \frac{\lambda}{n+\lambda}, \tag{52}$$

$$W_m^j = \frac{1}{2(n+\lambda)}, j = 1, ..., 2n, \tag{53}$$

$$W_c^0 = W_m^0 + (1 - \alpha^2 + \beta), \tag{54}$$

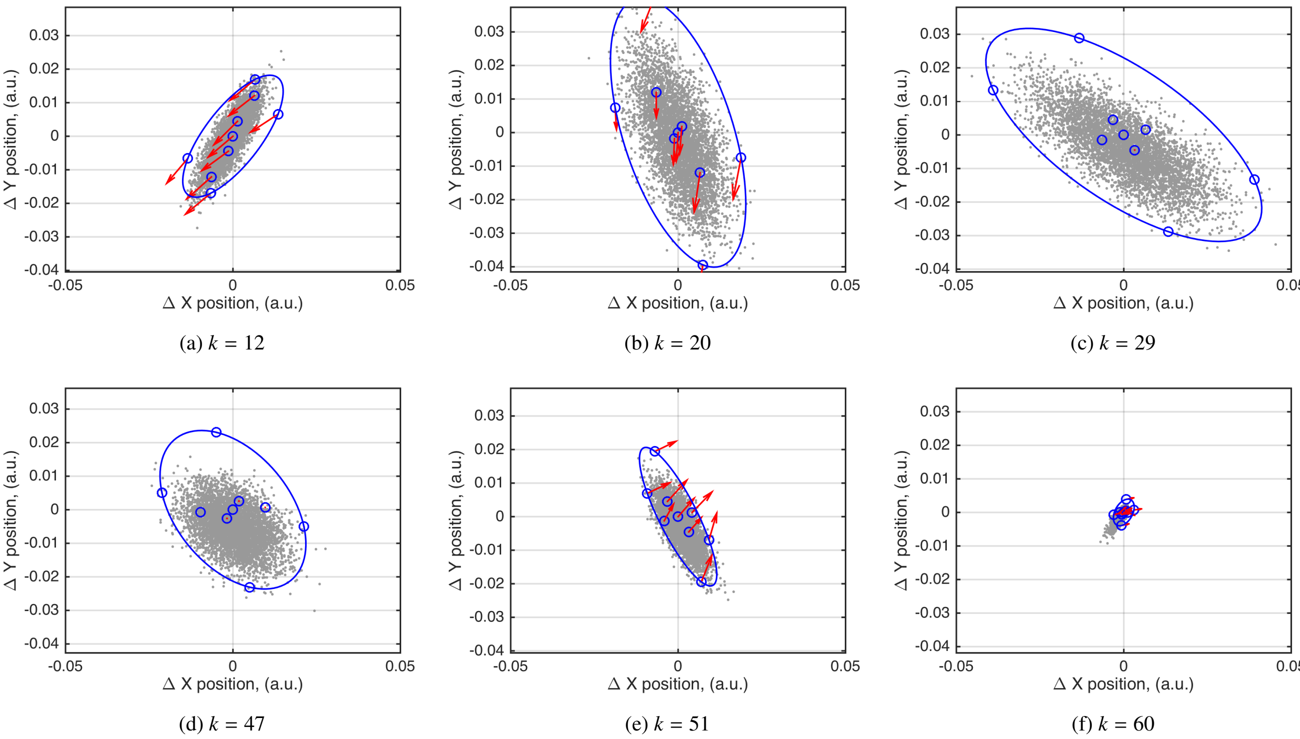
$$W_c^j = W_m^j, j = 1, ..., 2n, \tag{55}$$

and $\alpha \in (0, 1], \beta, \kappa \in (0, \infty)$ are arbitrary parameters. β should be 2.0 for Gaussian distribution.

Equations (46) to (48) derives the mapping φ_σ from $b[\boldsymbol{x}]$ to $\boldsymbol{X}$, and Eqs. (50) to (55) derives the mapping φ_b from $\boldsymbol{Y}$ to $b[\boldsymbol{y}]$ as

$$\boldsymbol{X} = \varphi_\sigma(b[\boldsymbol{x}]), \tag{56}$$

$$b[\boldsymbol{y}] = \varphi_b(\boldsymbol{Y}). \tag{57}$$



(a) $k = 12$ (b) $k = 20$ (c) $k = 29$

(d) $k = 47$ (e) $k = 51$ (f) $k = 60$

Fig. 8.: Evolution of stochastic process (k: stage, blue ellipse: 3-σ, blue circles: Sigma points).

Appendix B Dynamical System with A Priori Information

Given a priori information $\boldsymbol{X}_k \in \mathbb{R}^{n(2n+1)}$ and the control vector $\boldsymbol{U}_k \in \mathbb{R}^{m(2n+1)}$ corresponding to the state vectors $\boldsymbol{X}_k$, the propagated sigma points $\boldsymbol{X}_{k+1} \in \mathbb{R}^{n(2n+1)}$ are computed by

$$\boldsymbol{X}_{k+1} = \boldsymbol{F}_k(\boldsymbol{X}_k, \boldsymbol{U}_k, \boldsymbol{R}_k) \tag{58}$$

where $\boldsymbol{R}_k$ is the covariance matrix of disturbance vectors $\boldsymbol{w}_k$ and the nonlinear mapping $\boldsymbol{F}_k : \mathbb{R}^{n(2n+1)} \times \mathbb{R}^{m(2n+1)} \times \mathbb{R}^{n\times n} \to \mathbb{R}^{n(2n+1)}$ can be derived as follows.

Let us introduce the non-disturbed state vectors $\boldsymbol{z}_{k+1}$ and its sigma points $\boldsymbol{Z}_{k+1} = [\mathcal{Z}^0_{k+1}, ..., \mathcal{Z}^{2n}_{k+1}] \in \mathbb{R}^{n(2n+1)}$ as such that

$$\mathcal{Z}^0_{k+1} = f_k(\mathcal{X}^0_k, \mathcal{U}^0_k) \tag{59}$$

$$\mathcal{Z}^1_{k+1} = f_k(\mathcal{X}^1_k, \mathcal{U}^1_k) \tag{60}$$

$$\cdots \tag{61}$$

$$\mathcal{Z}^{2n}_{k+1} = f_k(\mathcal{X}^{2n}_k, \mathcal{U}^{2n}_k) \tag{62}$$

The mean value and covariance of the non-disturbed state vector $\boldsymbol{z}_{k+1}$ is derived by the nonlinear mapping φ_b as

$$\boldsymbol{Z}_{k+1} \xmapsto{\varphi_b} (\boldsymbol{\mu}_{k+1}, \tilde{\boldsymbol{P}}_{k+1}) := b[\boldsymbol{z}_{k+1}] \tag{63}$$

The mean value and covariance of the disturbed state vector $\boldsymbol{x}_{k+1}$ is therefore

$$b[\boldsymbol{x}_{k+1}] := (\boldsymbol{\mu}_{k+1}, \boldsymbol{P}_{k+1}) = (\boldsymbol{\mu}_{k+1}, \tilde{\boldsymbol{P}}_{k+1} + \boldsymbol{R}_k) \tag{64}$$

Finally, the nonlinear mapping φ_σ derives the propagated sigma points $\boldsymbol{X}_{k+1}$

$$b[\boldsymbol{x}_{k+1}] \xmapsto{\varphi_\sigma} \boldsymbol{X}_{k+1} \tag{65}$$

Appendix C Coefficients of Quadratic Form of TSDDP

The quadratically expanded Bellman's equation for TSDDP can be described as the quadratic form as shown in Eq.(29)

$$V^*_k(\boldsymbol{X}_k) = \min_{\delta \boldsymbol{U}_k} \left\{ Q_0 + \begin{bmatrix} \boldsymbol{Q}_X^T & \boldsymbol{Q}_U^T \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{X}_k \\ \delta\boldsymbol{U}_k \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \delta\boldsymbol{X}_k^T & \delta\boldsymbol{U}_k^T \end{bmatrix} \begin{bmatrix} \boldsymbol{Q}_{XX} & \boldsymbol{Q}_{XU} \\ \boldsymbol{Q}_{XU}^T & \boldsymbol{Q}_{UU} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{X}_k \\ \delta\boldsymbol{U}_k \end{bmatrix} \right\} \tag{66}$$

These coefficients are

$$Q_0 = \sum_{j=0}^{2n} W_m^j q_0^j \tag{67}$$

$$\boldsymbol{Q}_X = \begin{bmatrix} W_m^0 \boldsymbol{q}_x^0 \\ \vdots \\ W_m^{2n} \boldsymbol{q}_x^{2n} \end{bmatrix} \tag{68}$$

$$\boldsymbol{Q}_U = \begin{bmatrix} W_m^0 \boldsymbol{q}_u^0 \\ \vdots \\ W_m^{2n} \boldsymbol{q}_u^{2n} \end{bmatrix} \tag{69}$$

$$\boldsymbol{Q}_{XX} = \begin{bmatrix} W_m^0 \boldsymbol{q}_{xx}^0 & & \boldsymbol{O} \\ & \ddots & \\ \boldsymbol{O} & & W_m^{2n} \boldsymbol{q}_{xx}^{2n} \end{bmatrix} \tag{70}$$

$$\boldsymbol{Q}_{XU} = \begin{bmatrix} W_m^0 \boldsymbol{q}_{xu}^0 & & \boldsymbol{O} \\ & \ddots & \\ \boldsymbol{O} & & W_m^{2n} \boldsymbol{q}_{xu}^{2n} \end{bmatrix} \tag{71}$$

$$\boldsymbol{Q}_{UU} = \begin{bmatrix} W_m^0 \boldsymbol{q}_{uu}^0 & & \boldsymbol{O} \\ & \ddots & \\ \boldsymbol{O} & & W_m^{2n} \boldsymbol{q}_{uu}^{2n} \end{bmatrix} \tag{72}$$

and

$$q_0^j = L_{k,0} + A_{k+1} + \boldsymbol{B}_{k+1}^T \boldsymbol{f}_{k,0} + \frac{1}{2}\boldsymbol{f}_{k,0}^T \boldsymbol{C}_{k+1} \boldsymbol{f}_{k,0} \quad (73)$$

$$\boldsymbol{q}_x^{jT} = \boldsymbol{L}_{x,k}^T + (\boldsymbol{B}_{k+1}^T + \boldsymbol{f}_{k,0}^T \boldsymbol{C}_{k+1}) \boldsymbol{f}_{x,k}^T \quad (74)$$

$$\boldsymbol{q}_u^{jT} = \boldsymbol{L}_{u,k}^T + (\boldsymbol{B}_{k+1}^T + \boldsymbol{f}_{k,0}^T \boldsymbol{C}_{k+1}) \boldsymbol{f}_{u,k}^T \quad (75)$$

$$\boldsymbol{q}_{xx}^j = \boldsymbol{L}_{xx,k} + (\boldsymbol{B}_{k+1}^T + \boldsymbol{f}_{k,0}^T \boldsymbol{C}_{k+1}) \star \boldsymbol{f}_{xx,k} + \boldsymbol{f}_{x,k} \boldsymbol{C}_{k+1} \boldsymbol{f}_{x,k}^T \quad (76)$$

$$\boldsymbol{q}_{xu}^j = \boldsymbol{L}_{xu,k} + (\boldsymbol{B}_{k+1}^T + \boldsymbol{f}_{k,0}^T \boldsymbol{C}_{k+1}) \star \boldsymbol{f}_{xu,k} + \boldsymbol{f}_{x,k} \boldsymbol{C}_{k+1} \boldsymbol{f}_{u,k}^T \quad (77)$$

$$\boldsymbol{q}_{uu}^j = \boldsymbol{L}_{uu,k} + (\boldsymbol{B}_{k+1}^T + \boldsymbol{f}_{k,0}^T \boldsymbol{C}_{k+1}) \star \boldsymbol{f}_{uu,k} + \boldsymbol{f}_{u,k} \boldsymbol{C}_{k+1} \boldsymbol{f}_{u,k}^T \quad (78)$$

where the operator $\star$ is defined as $(\boldsymbol{a} \star \boldsymbol{b})_{ij} = \sum_p a_p b_{ijp}$ and

$$A_{k+1} = V_{k+1,0}^* + \sum_{p=0}^{2n} \boldsymbol{\mathcal{V}}_{k+1,x}^{pT} \left(\boldsymbol{\mathcal{W}}_k^p - \bar{\boldsymbol{X}}_{k+1}^p \right) + \sum_{p=0}^{2n} \sum_{q=0}^{2n} \left(\boldsymbol{\mathcal{W}}_k^{pT} - \bar{\boldsymbol{X}}_{k+1}^{pT} \right) \boldsymbol{\mathcal{V}}_{k+1,xx}^{pq} \left(\boldsymbol{\mathcal{W}}_k^q - \bar{\boldsymbol{X}}_{k+1}^q \right)$$

$$\boldsymbol{B}_{k+1} = \sum_{p=0}^{2n} \boldsymbol{\mathcal{V}}_{k+1,x}^p + \sum_{p=0}^{2n} \sum_{q=0}^{2n} \left(\boldsymbol{\mathcal{W}}_k^{pT} - \bar{\boldsymbol{X}}_{k+1}^{pT} \right) \boldsymbol{\mathcal{V}}_{k+1,xx}^{pq} \quad (79)$$

$$\boldsymbol{C}_{k+1} = \sum_{p=0}^{2n} \sum_{q=0}^{2n} \boldsymbol{\mathcal{V}}_{k+1,xx}^{pq} \quad (80)$$

AppendixD Constrained Differential Dynamic ProgrammingAlgorithm

Let us find the optimal linear control policy $\delta \boldsymbol{U}_k^* = \boldsymbol{\alpha}_k + \boldsymbol{\beta}_k \delta \boldsymbol{X}_k$ by solving the following constrained quadratic programming

$$\min_{\delta \boldsymbol{U}_k} \left\{ Q_0 + \begin{bmatrix} \boldsymbol{Q}_X^T & \boldsymbol{Q}_U^T \end{bmatrix} \begin{bmatrix} \delta \boldsymbol{X}_k \\ \delta \boldsymbol{U}_k \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \delta \boldsymbol{X}_k^T & \delta \boldsymbol{U}_k^T \end{bmatrix} \begin{bmatrix} \boldsymbol{Q}_{XX} & \boldsymbol{Q}_{XU} \\ \boldsymbol{Q}_{XU}^T & \boldsymbol{Q}_{UU} \end{bmatrix} \begin{bmatrix} \delta \boldsymbol{X}_k \\ \delta \boldsymbol{U}_k \end{bmatrix} \right\} \quad (81)$$

subject to

$$\boldsymbol{G}_0 + \boldsymbol{G}_U \delta \boldsymbol{U}_k \leq \boldsymbol{0} \quad (82)$$

Murray et.al.[28] and Yakowitz[29] have proposed the algorithm to solve this problem. The first step solves the quadratic programming with assuming $\delta \boldsymbol{X}_k = \boldsymbol{0}$

$$\min_{\delta \boldsymbol{U}_k} \left\{ \boldsymbol{Q}_U^T \delta \boldsymbol{U}_k + \delta \boldsymbol{U}_k^T \boldsymbol{Q}_{UU} \delta \boldsymbol{U}_k \right\} \quad (83)$$

subject to

$$\boldsymbol{G}_0 + \boldsymbol{G}_U \delta \boldsymbol{U}_k \leq \boldsymbol{0} \quad (84)$$

The second step evaluates whether the constraints are active or inactive, and assume that the active constraints will be active even if we consider $\delta \boldsymbol{X}_k \neq \boldsymbol{0}$, and vice versa. Therefore, we can obtain the optimal linear control policy $\delta \boldsymbol{U}_k^* = \boldsymbol{\alpha}_k + \boldsymbol{\beta}_k \delta \boldsymbol{X}_k$ from

$$\min_{\delta \boldsymbol{U}_k} \left\{ Q_0 + \begin{bmatrix} \boldsymbol{Q}_X^T & \boldsymbol{Q}_U^T \end{bmatrix} \begin{bmatrix} \delta \boldsymbol{X}_k \\ \delta \boldsymbol{U}_k \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \delta \boldsymbol{X}_k^T & \delta \boldsymbol{U}_k^T \end{bmatrix} \begin{bmatrix} \boldsymbol{Q}_{XX} & \boldsymbol{Q}_{XU} \\ \boldsymbol{Q}_{XU}^T & \boldsymbol{Q}_{UU} \end{bmatrix} \begin{bmatrix} \delta \boldsymbol{X}_k \\ \delta \boldsymbol{U}_k \end{bmatrix} \right\} \quad (85)$$

subject to

$$\hat{\boldsymbol{G}}_0 + \hat{\boldsymbol{G}}_U \delta \boldsymbol{U}_k = \boldsymbol{0} \quad (86)$$

where $\hat{\boldsymbol{G}}_0$ and $\hat{\boldsymbol{G}}_u$ are that for active constraints.

References

1) John T. Betts. Survey of numerical methods for trajectory optimization. *Journal of Guidance, Control and Dynamics*, 21(2):193–207, 1998.
2) Bruce A. Conway. *Spacecraft Trajectory Optimization*. Cambridge Aerospace Series. Cambridge University Press, 2010.
3) Gregory Whiffen. Static/dynamic control for optimizing a useful objective, 12 2003.
4) Camilla Colombo, Massimiliano Vasile, and Gianmarco Radice. Optimal low-thrust trajectories to asteroids through an algorithm based on differential dynamic programming. *Celestial Mechanics and Dynamical Astronomy*, 105(1):75, 2009.
5) Gregory Lantoine and Ryan P. Russell. A Hybrid Differential Dynamic Programming Algorithm for Constrained Optimal Control Problems. Part 1: Theory. *Journal of Optimization Theory and Applications*, 154(2):382–417, apr 2012.
6) Gregory Lantoine and Ryan P. Russell. A Hybrid Differential Dynamic Programming Algorithm for Constrained Optimal Control Problems. Part 2: Application. *Journal of Optimization Theory and Applications*, 154(2):418–442, apr 2012.
7) Joris T. Olympio. A continuous implementation of a second-variation optimal control method for space trajectory problems. *Journal of Optimization Theory and Applications*, 158(3):687–716, 2013.
8) David H. Jacobson and David Q. Mayne. *Differential dynamic programming*. American Elsevier Pub. Co., 1970.
9) Marc D. Rayman, Thomas C. Fraschetti, Carol A. Raymond, and Christopher T. Russell. Coupling of system resource margins through the use of electric propulsion: Implications in preparing for the dawn mission to ceres and vesta. *Acta Astronautica*, 60(1011):930 – 938, 2007.
10) Joris T. Olympio and Chit Hong Yam. Deterministic method for space trajectory design with mission margin constraints. In *61th International Astronautical Congress*, 2010.
11) Joris T. Olympio. Designing robust low-thrust interplanetary trajectories subejct to one temporary engine failure. In *20th AAS/AIAA Space Flight Mechanics*, San Diego, CA, USA, 2010.
12) L. Chisci, J. a. Rossiter, and G. Zappa. Systems with persistent disturbances: Predictive control with restricted constraints. *Automatica*, 37(7):1019–1028, 2001.
13) A. Richards and J. How. Robust stable model predictive control with constraint tightening. In *2006 American Control Conference*, pages 6 pp.–, June 2006.
14) P. O. M. Scokaert and D. Q. Mayne. Min-max feedback model predictive control for constrained linear systems. *IEEE Transactions on Automatic Control*, 43(8):1136–1142, Aug 1998.
15) M. Diehl and J. Bjornberg. Robust dynamic programming for min-max model predictive control of constrained uncertain systems. *IEEE Transactions on Automatic Control*, 49(12):2253–2257, Dec 2004.
16) A. Bemporad, F. Borrelli, and M. Morari. Min-max control of constrained uncertain discrete-time linear systems. *IEEE Transactions on Automatic Control*, 48(9):1600–1606, Sept 2003.
17) W. Langson, I. Chryssochoos, S.V. Rakovi, and D.Q. Mayne. Robust model predictive control using tubes. *Automatica*, 40(1):125 – 133, 2004.
18) Evangelos Theodorou, Yuval Tassa, and Emo Todorov. Stochastic Differential Dynamic Programming. *Proceedings of the 2010 American Control Conference*, 0(1):1125–1132, jun 2010.
19) Emanuel Todorov and Yuval Tassa. Iterative local dynamic programming. *Adaptive Dynamic Programming and Reinforcement Learning*, pages 0–5, 2009.
20) Yunpeng Pan and Evangelos Theodorou. Probabilistic differential dynamic programming. In *Advances in Neural Information Processing Systems 27*, pages 1907–1915. Curran Associates, Inc., 2014.
21) Simon J Julier and Jeffrey K Uhlmann. New extension of the kalman filter to nonlinear systems. In *AeroSense'97*, pages 182–193. International Society for Optics and Photonics, 1997.

22) I. Michael Ross, Ronald J. Proulx, and Mark Karpenko. Unscented Optimal Control for Space Flight. In *24th International Symposium on Space Flight Dynamics (ISSFD)*, pages 1–12, Laurel, MD, 2014.
23) I. Michael Ross, Ronald J. Proulx, and Mark Karpenko. Unscented guidance. *Proceedings of the American Control Conference*, 2015-July:5605–5610, 2015.
24) Zachary Manchester and Scott Kuindersma. Derivative-free trajectory optimization with unscented dynamic programming. In *Proceedings of the 55th Conference on Decision and Control (CDC)*, 2016.
25) Naoya Ozaki, Ryu Funase, Stefano Campagnola, and Chit Hong Yam. Stochastic Differential Dynamic Programming for Low-Thrust Trajectory Design with State Uncertainty. In *26th AAS/AIAA Space Flight Mechanics Meeting*, 2016.
26) Jiongmin Yong and Xun Yu Zhou. *Stochastic Controls - Hamiltonian Systems and HJB Equations*. Springer-Verlag New York, Inc., 1999.
27) R.E. Bellman and S.E. Dreyfus. *Applied dynamic programming*. Princeton University Press, Princeton, 1962.
28) Daniel M. Murray and Sidney J. Yakowitz. Constrained differential dynamic programming and its application to multireservoir control. *Water Resources Research*, 15(5):1017, 1979.
29) S.J. Yakowitz. The stagewise Kuhn-Tucker condition and differential dynamic programming. *IEEE Transactions on Automatic Control*, (I):25–30, 1986.
30) Oliver Montenbruck and Eberhard Gill. *Satellite Orbits - Models, Methods, and Applications*. Springer-Verlag Berlin Heidelberg, 2000.
31) Nagavenkat Adurthi, Puneet Singla, and Manoranjan Majji. Sparse ApproximationBased Collocation Scheme for Nonlinear Optimal Feedback Control Design. *Journal of Guidance, Control, and Dynamics*, 40(2):248–264, 2017.