Low-Thrust Trajectory Design to Improve Overall Mission Success Probability Incorporating Target Changes in Case of Engine Failures

By Akifumi WACHI,[1)]

[1)]*Department of Aeronautics and Astronautics, The University of Tokyo, Tokyo, Japan*

(Received February 1st, 2017)

This paper presents a method to design robust deep space trajectories with the capability to change the target celestial body in case of failures of low-thrust engines. Increasing number of deep space spacecraft targeting various celestial bodies using low thrust engines have been developed and launched. As such low thrust engines should operate for a long time during cruise phase, it is probable that some of the implemented engines may fail. Conventional trajectory design methods have pursued fuel minimum solution to reach the main target celestial body, and it would usually be very difficult to change the target to the second candidate even in the case of failure of engines. As a result, when a failure occurs in the engines, the spacecraft cannot reach any celestial bodies at all. In order to avoid such situation, it would be desirable if the spacecraft can easily change the targets in case of engine failures. This paper proposes a design method of low thrust trajectory which allows target change more easily at a cost of less optimized trajectory to the main target. In this research, the objective function is defined as weighted sum of probability to reach the main and backup targets, which is optimized in a stochastic way. By utilizing sampling-based algorithm and Bellman equation, the proposed method can efficiently extend trees so that an approximately optimal solution can be found with small computational time. Finally, the numerical simulation is shown, which demonstrates that a proposed algorithm improves the mission success probability to reach either of main or backup targets.

Key Words: Trajectory Design, Fault Tolerance, Backup Trajectory, Dynamic Programming, Sampling-Based Algorithm

Nomenclature

$\boldsymbol{x}$	:	discrete-time state vector
$\boldsymbol{r}$	:	discrete-time position vector
$\boldsymbol{v}$	:	discrete-time velocity vector
$\boldsymbol{u}$	:	discrete-time control input vector
$\boldsymbol{\pi}$	:	optimal control policy
w	:	exploration worth
J	:	objective function
V	:	value function
n	:	the number of normal thrusters
m	:	the number of target candidates
Φ	:	State Transition Matrix (STM)
P	:	transition probability on the number of normal thrusters
$\wedge$	:	Logical AND (conjunction)
$\vee$	:	Logical OR (disjunction)
Subscripts		
k	:	decision stage
N	:	the number of decision stages
Superscript		
j	:	label of celestial body

Introduction

Low-thrust propulsion systems have received considerable attention for space missions since we can efficiently achieve large delta-V by adopting them. Because of low reliability of low-thrust engines, however, most of the space probes with the engines have experienced engine failures. Since thrusters failures are likely to result in mission failure, it is significant to be sensitive to the risk of engine failures. Based on this background, this research aims to realize fault-tolerance by intelligently designing a trajectory. In order to accomplish this purpose, we consider the probability of the engine permanent failure and then introduce the flexibility that backup trajectories can be chosen as additional decision variables. This paper presents the robust trajectory design method by choosing backup trajectories when the permanent failure of low-thrust engines occurs.

Backup policy is a quite familiar concept in our daily lives. When pilots of aircraft are in the trouble of engine failures, they often choose a backup policy for the safety e.g. safe emergency landing or precautionary landing. Strict manuals have been prepared considering possible troubles, and pilots take measures suited to the occasion. Since the situation becomes exacerbated in the case of engine failures, pilots cannot but give up persisting in the primary objective. For example, a pilot, leaving from Paris, must make a decision to emergently target at London instead of New York in the case of trouble. In this example, New York is the primary target, and a pilot adopts a backup policy towards London due to the trouble of engine failure. Aircraft are designed to work for more than ten years with the higher reliability than spacecraft, space engineers should be more sensitive to engine failures and considerate of backup policy.

As to space engineering which is highly conservative to improve reliability, previous works have maintained the robustness by intelligently building redundant systems. A great deal of past work[1)2)3)] realized robustness or survivability against unfavorable probabilistic events in terms of system design. Our research is based on the similar concept in the sense that we also focus on the failure possibility, but our method maintains it in terms of trajectory design. Recently, there has been an increasing number of work on the typical problems of the low-thrust trajectory design. The issues especially addressed in recent

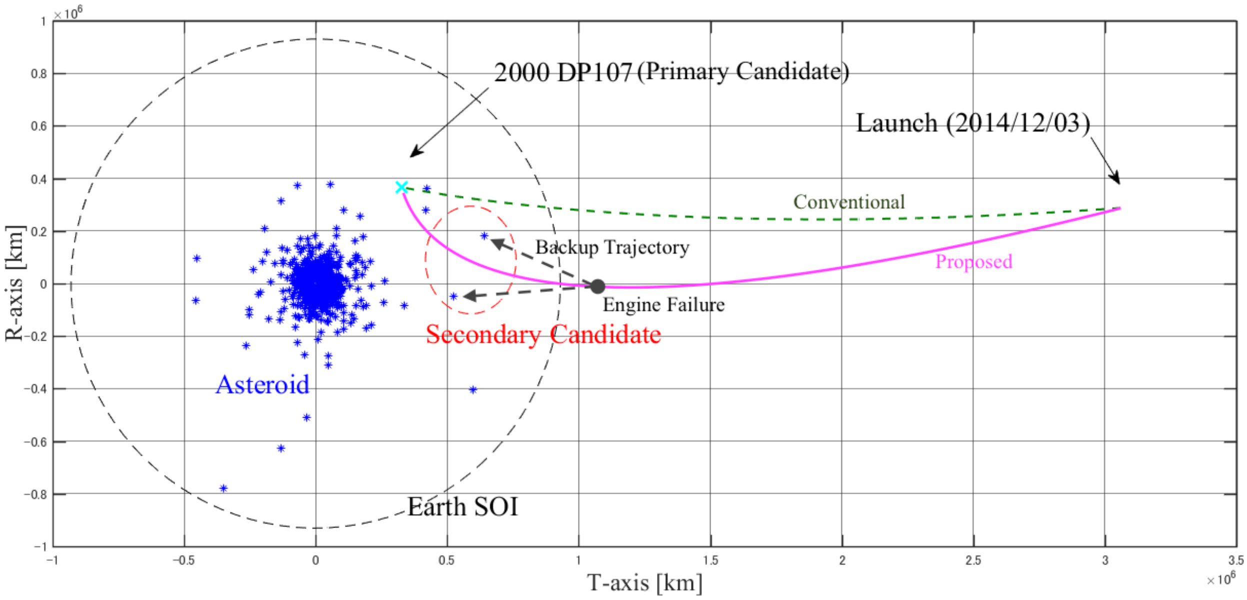

Fig. 1. Concept of this study.

years are possible underperformance of the engines or missed thrust.[4)5)] A great deal of past work has addressed the issue of temporal failure such as missed thrust, but we also should keep sensitive against the risk of permanent failure of low-thrust propulsion systems. Most of the trajectory design method including low-thrust trajectory design has solved optimization problem defining the objective function as fuel consumption or orbit transfer time[6)7)]. Also, previous work solved the multi-objective optimization problem, focusing on both of the final spacecraft mass and time-of-flight.[8)9)] However, it is not essential objective for space missions to minimize fuel consumption or orbit transfer time. On the other hand, especially in the field of robotics, the objective function directly expressing the mission accomplishment is defined and optimized. The previous research defined the cost function coming from a weighted sum of threat and path length.[10)] Also, an algorithm named Stochastic Motion Roadmap (SMRM) is proposed to generate the path maximizing the probability of avoiding collisions and successfully reaching the goal.[11)] Furthermore, optimal control problems with stochastic constraint, Chance-Constrained Optimal Control (CCOC) problems, have been well studied.[12)] In this control theory, path planning or motion planning can be conducted with considering the possibility of risk or failure.

Although the trajectory design community has tackled a problem on the risk or failure, most of the previous work takes into account only short-term failure, missed-thrust of low-thrust propulsion systems. On the other hand, this research considers permanent failures of electrically powered spacecraft propulsion system. In order to maintain the robustness against the failure of low thrust engines, we define the objective function as the weighted sum of possibility to reach, which can directly optimize the essential objective for space missions. This objective function allows us to take into account more than one target celestial bodies and include the idea of backup trajectories. Figure 1 describes the concept of this research. This figure indicates a real trajectory on B plane of the deep space probe PROCYON.[13)] The origin of B-plane is the Earth, and blue dots are positions of asteroids on B-plane. Conventional methods (green dashed line) is a trajectory adopted in PROCYON project, which has pursued fuel minimum solution to the primary candidate. However, proposed trajectory (magenta) are more likely to enable the space probe to accomplish the worthwhile missions by changing the target celestial object even if engine failure occurs. The author's previous work[14)] showed preliminary results and underlies this paper. The previous paper proposed a method of trajectory design with multiple target celestial bodies, but theoretical discussion on the optimality or connection with the conventional method is immature. In this paper, the object function is defined as expected worth of mission accomplishment. Theoretical analysis proves that this objective function is so compatible that it can include conventional one. Optimal trajectory is searched by our proposed method, a novel sampling method which gains efficiency by applying the Bellman equation. This proposed method is found to have some favorable property of probabilistic completeness by theoretical study, Finally, the numerical simulation demonstrated that a proposed method improved the mission achievement.

1. Problem Statement

This paper considers the problem of finite-horizon optimal control of dynamic systems, with state and control constraints. We assume a discrete-time, continuous-state dynamics model. Let $\mathcal{X}$ be the state space. The state vector $\boldsymbol{x}_k \in \mathcal{X}$ consists of the position and velocity written as

$$\boldsymbol{x}_k := [\, \boldsymbol{r}_k^\top,\ \boldsymbol{v}_k^\top]^\top,$$

where $\boldsymbol{r} \in \mathbb{R}^3$ is the position vector, and $\boldsymbol{v} \in \mathbb{R}^3$ is the velocity vector. The subscript $k\ (= 1, 2, \cdots, N-1)$ indicates the decision stage i.e. time. The state of the system $\boldsymbol{x}_k$ is fully observable at all times. Let $\boldsymbol{u}_k \in \mathcal{U}_k(\boldsymbol{x}_k) \subset \mathcal{U}$ denote the control input by thrusters, where $\mathcal{U}_k(\boldsymbol{x}_k)$ is the state-dependent control space, and $\mathcal{U}$ is the control space. Consider the time-variant system and next state vector $\boldsymbol{x}_{k+1}$ is given by the following equation:

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}_k(\boldsymbol{x}_k, \boldsymbol{u}_k), \quad \forall k \in [1, N-1],$$

where $f : [1\ N-1] \times \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^3$ is a given function.

The objective function to maximize in this problem is dependent on the number of normal engines as well as the state vector. When a engine is not failed or broken, it is called *normal engine*, hereafter. Let $n_k \in \mathbb{Z}$ denote the number of normal engines, which is also fully observable at all times. The number of engines is assumed to be a positive integer. In other words, indecisive or temporary failures are not considered in this paper. No engine is assumed to fail at stage 1, and then n_1 means the number of mounted engines $n_{\max} \in \mathbb{Z}$. The objective function to maximize in this research is defined as

$$J_k(\boldsymbol{x}_k, n_k, \boldsymbol{u}_{k:N-1}) := \sum_{j=1}^{m} w_j \Pr\Big[\boldsymbol{r}_N = \boldsymbol{r}^j \mid \boldsymbol{x}_k, n_k\Big], \qquad (1)$$

where $\boldsymbol{u}_{k:N-1}$ is a sequence of control and defined as

$$\boldsymbol{u}_{k:N-1} := \{\boldsymbol{u}_k(\boldsymbol{x}_k, n_k), \cdots \boldsymbol{u}_{N-1}(\boldsymbol{x}_{N-1}, n_{N-1})\}.$$

$w_j \in \mathbb{R}$ is the exploration worth of j-th candidate. $\boldsymbol{r}^j \in \mathbb{R}^3$ is the position of j-th candidate at terminal time. $\Pr[\boldsymbol{r}_N = \boldsymbol{r}^j \mid \boldsymbol{x}_k, n_k]$ is the possibility that the space probe can arrive at j-th candidate under the condition of state $\boldsymbol{x}_k$ and the number of normal engines n_k. $m \in \mathbb{Z}$ is the number of candidates to consider in this problem. We call the first candidate (i.e. $w_1 > w_j,\ j = 2, \cdots, m$) *Primary Candidate (P.C.)*. Therefore, the optimal control policy

$$\boldsymbol{\pi}_k(\boldsymbol{x}_k, n_k) := \{\boldsymbol{u}_k^*(\boldsymbol{x}_k, n_k), \cdots, \boldsymbol{u}_{N-1}^*(\boldsymbol{x}_{N-1}, n_{N-1})\}$$

to maximize the objective function $J_k(\boldsymbol{x}_k, n_k, \boldsymbol{u}_{k:N-1})$ should be searched such that

$$\boldsymbol{\pi}_k(\boldsymbol{x}_k, n_k) := \underset{\boldsymbol{u}_{k:N-1} \in \boldsymbol{\mathcal{U}}_{k:N-1}(\boldsymbol{x}_k, n_k)}{\arg\max} J_k(\boldsymbol{x}_k, n_k, \boldsymbol{u}_{k:N-1}), \qquad (2)$$

where $\boldsymbol{\mathcal{U}}_{k:N-1}(\boldsymbol{x}_k, n_k)$ is the admissible state dependent control set from stage k to stage $N-1$. Independent variables of admissible control set must be defined as the number of normal engines n_k as well as $\boldsymbol{x}_k$; That is, the admissible control set is denoted as $\boldsymbol{\mathcal{U}}_{k:N-1}(\boldsymbol{x}_k, n_k)$ instead of $\boldsymbol{\mathcal{U}}_{k:N-1}(\boldsymbol{x}_k)$. The finite-horizon optimal control problem for the spacecraft at stage k is formally stated as follows.

$$\begin{aligned}
\max &: J_k(\boldsymbol{x}_k, n_k, \boldsymbol{u}_{k:N-1}) = \sum_{j=1}^{m} w_j \Pr\Big[\boldsymbol{r}_N = \boldsymbol{r}^j \mid \boldsymbol{x}_k, n_k\Big] \\
\text{find} &: \boldsymbol{\pi}_k(\boldsymbol{x}_k, n_k) = \{\boldsymbol{u}_k^*(\boldsymbol{x}_k, n_k), \cdots, \boldsymbol{u}_{N-1}^*(\boldsymbol{x}_{N-1}, n_{N-1})\} \\
\text{s.t.} &: \boldsymbol{x}_{i+1} = \boldsymbol{f}_i(\boldsymbol{x}_i, \boldsymbol{u}_i)\ \ \forall i \in [k, N-1] \\
&\quad \boldsymbol{u}_i(\boldsymbol{x}_i, n_i) \in \boldsymbol{\mathcal{U}}_i(\boldsymbol{x}_i, n_i) \subseteq \boldsymbol{\mathcal{U}}_i(\boldsymbol{x}_i, n_{\max}) \subseteq \boldsymbol{\mathcal{U}}
\end{aligned}$$

2. Markov Decision Processes (MDPs)

It is almost impossible to derive the (approximately) optimal policy without any assumption. Due to the high-dimensional and continuous state space, it is almost intractable to solve this optimal control problem in a reasonable computational time. Therefore, we assume first-order Markov property and model this problem as Markov Decision Processes (MDPs). For an introduction to MDPs, we refer the reader to Sutton & Barto (1998)[15] or Bertsekas & Tsitsiklis (1996).[16] Low-thrust engines are assumed to fail depending on the norm of control input i.e. $\|\boldsymbol{u}_k\|$ at each segment. Assuming first-order Markov property, the transition probability on the number of normal engines can be defined as $P(n_{k+1} \mid \boldsymbol{u}_k, n_k)$.

A MDP can be defined as tuple $< \mathcal{X}, \mathcal{N}, \mathcal{A}, f(\boldsymbol{x}, \boldsymbol{u}), \mathcal{P} >$. $\mathcal{X}$ is a set of states $\boldsymbol{x}$, $\mathcal{N}$ is a set of the number of normal engines n, $\mathcal{A}$ is a set of actions u, $f(\boldsymbol{x}, \boldsymbol{u})$ is a deterministic state transition, and $P : \mathcal{N} \times \mathcal{A} \times \mathcal{N} \to \mathbb{R}$ is the transition probability on the number of normal engines. The objective function at stage k, $J_k(\boldsymbol{x}_k, n_k)$ satisfies the following recurrence relation.

$$\begin{aligned}
&J_k(\boldsymbol{x}_k, n_k, \boldsymbol{u}_{k:N-1}) \\
&\quad = \int_{\mathbb{Z}} P(n_{k+1} \mid \boldsymbol{u}_k, n_k) \cdot J_{k+1}(\boldsymbol{x}_{k+1}, n_{k+1}, \boldsymbol{u}_{k+1:N-1})\, dn_{k+1}.
\end{aligned}$$

When only permanent engine failures are considered, we can use summation instead of intergral, and the following equation holds:

$$J_k(\boldsymbol{x}_k, n_k, \boldsymbol{u}_{k:N-1}) = \sum_{i=0}^{n_{\max}} P(i \mid \boldsymbol{u}_k, n_k) \cdot J_{k+1}(\boldsymbol{x}_{k+1}, i, \boldsymbol{u}_{k+1:N-1}).$$

If the situation that failed engines get fixed, is not considered, transition probability should be defined as:

$$P(j \mid \boldsymbol{u}_k, n_k) = 0, \quad \forall j \in [n_k + 1, n_{\max}].$$

In case of stage N, the objective function is defined as

$$J_N(\boldsymbol{x}_N, n_N) = \begin{cases} w_j & \text{if } (\boldsymbol{r}_N = \boldsymbol{r}^j) \land (n_N \geq 1) \\ 0 & \text{otherwise} \end{cases} \quad \forall j \in [1, m].$$

By employing Bellman's principle of optimality,[17][18] the following recurrence relation on the value function is satisfied:

$$\begin{aligned}
&V_k(\boldsymbol{x}_k, n_k) \\
&\quad = \max_{\boldsymbol{u}_k} \int_{\mathbb{Z}} P(n_{k+1} \mid \boldsymbol{u}_k, n_k) \cdot V_{k+1}(\boldsymbol{x}_{k+1}, n_{k+1})\, dn_{k+1},
\end{aligned} \qquad (3)$$

where $V_k(\boldsymbol{x}_k, n_k)$ is the value function and expressed as

$$V_k(\boldsymbol{x}_k, n_k) := \max_{\boldsymbol{u}_{k:N-1}} J_k(\boldsymbol{x}_k, n_k, \boldsymbol{u}_{k:N-1}).$$

If the possibility that failed engines get fixed is taken into account, Equation 3 is described as

$$V_k(\boldsymbol{x}_k, n_k) = \sum_{i=0}^{n_{\max}} P(i \mid \boldsymbol{u}_k, n_k) \cdot V_{k+1}(\boldsymbol{x}_{k+1}, i).$$

Also, if the situation that failed engines get fixed is not considered, Equation 3 is described as

$$V_k(\boldsymbol{x}_k, n_k) = \sum_{i=0}^{n_k} P(i \mid \boldsymbol{u}_k, n_k) \cdot V_{k+1}(\boldsymbol{x}_{k+1}, i).$$

Value function for stage N is defined as

$$V_N(\boldsymbol{x}_N, n_N) = \begin{cases} w_j & \text{if } (\boldsymbol{r}_N = \boldsymbol{r}^j) \land (n_N \geq 1) \\ 0 & \text{otherwise} \end{cases} \quad \forall j \in [1, m].$$

Equations 3 has a recursive form on the value function function, which is generally called Bellman equation.

3. Theoretical Study on Problem Formulation

A great deal of previous studies have worked on the trajectory design problem with only one target celestial body. On the other hand, this paper deals with the trajectory design which takes into account more than one thrusters and target celestial bodies. Our objective function is defined as the weighted sum of possibility to reach each celestial body, which seems to be conflicting with the conventional one e.g. fuel minimum consumption. However, the problem statement is compatible with the conventional, and then fuel minimum solution can be obtained by optimizing the objective function. That is, our proposed method can include the conventional one. In this section, we show that fuel minimum solution is optimal in case that only one normal engine is normal or one celestial body is targeted.

Theorem 1. *When a spacecraft targets at only one celestial body, optimal policy is to choose the fuel minimum solution towards the celestial body.*

Proof. When only one celestial body is considered, the object function is defined as

$$J_k(\boldsymbol{x}_k, n_k) = w_1 \cdot \Pr[r_N = r^1 \mid \boldsymbol{x}_k, n_k].$$

Possibility to reach corresponds to the probability that not all engines have failed until stage N, so

$$\begin{aligned}\Pr[r_N = r^1 \mid \boldsymbol{x}_k, n_k] &= 1 - \prod_{l=1}^{n_k} P(0 \,|\, \boldsymbol{u}^l_{k:N-1}, 1) \\ &= 1 - \prod_{l=1}^{n_k} (1 - \prod_{i=k}^{N-1} P(1 \,|\, \boldsymbol{u}^l_i, 1)) \\ &= 1 - \prod_{l=1}^{n_k} (1 - \prod_{i=k}^{N-1} (1-p)^{\frac{\|\boldsymbol{u}^l_i\|}{\bar{u}}}) \\ &= 1 - \prod_{l=1}^{n_k} (1 - (1-p)^{\frac{1}{\bar{u}} \sum_{i=k}^{N-1} \|\boldsymbol{u}^l_i\|}),\end{aligned}$$

where $\boldsymbol{u}^l_{k:N-1}$ is the control set of l-th engine from stage k to stage $N-1$. In addition, $\boldsymbol{u}_i$ is admissible control to arrive at the candidate. Let $q \in \mathbb{R}$ and $\boldsymbol{U}^l_{k:N-1}$ denote $q = (1-p)^{\frac{1}{\bar{u}}}$ and $\boldsymbol{U}^l_{k:N-1} = \sum_{i=k}^{N-1} \|\boldsymbol{u}^l_i\|$. Probability to reach and the object function is simply expressed as

$$\Pr[r_N = r^1 \mid \boldsymbol{x}_k, n_k] = 1 - \prod_{l=1}^{n_k} (1 - q^{\boldsymbol{U}^l_{k:N-1}}).$$

Consider a solution whose fuel consumption is more than the fuel minimum solution. Let the increment of fuel consumption denote $\varepsilon \in \mathbb{R}$, $\varepsilon \geq 0$. Describe the fuel minimum consumption of l-th engine from stage k to stage $N-1$ as $\tilde{\boldsymbol{U}}^l_{k:N-1}$ When the increase of the fuel consumption, ε is forced to $\hat{l}$-th engine, the object function is presented as

$$\begin{aligned}\Pr[r_N = r^1 \mid \boldsymbol{x}_k, n_k] &= 1 - \Big\{ \prod_{l=1,\, l \neq \hat{l}}^{n_k} (1 - q^{\tilde{\boldsymbol{U}}^l_{k:N-1}}) \Big\} \cdot (1 - q^{\tilde{\boldsymbol{U}}^{\hat{l}}_{k:N-1} + \varepsilon}) \\ &= 1 - \Big\{ \prod_{l=1,\, l \neq \hat{l}}^{n_k} (1 - q^{\tilde{\boldsymbol{U}}^l_{k:N-1}}) \Big\} \cdot (1 - q^{\tilde{\boldsymbol{U}}^{\hat{l}}_{k:N-1}} \cdot q^{\varepsilon}) \\ &\leq 1 - \prod_{l=1}^{n_k} (1 - q^{\tilde{\boldsymbol{U}}^l_{k:N-1}}).\end{aligned}$$

We can also derive the same theoretical result as to a situation that increase of the fuel consumption ε is forced to more than one engines.

In the end, it is proved that optimal policy is to choose the fuel minimum solution when a space probe targets at only one celestial body. □

Theorem 2. *When a spacecraft has only one normal engine, optimal policy is to choose the fuel minimum solution towards the celestial body which has the maximum expected value of exploration. That is, when only one engine is normal, value function is easily denoted as following equation.*

$$\begin{aligned}V_k(\boldsymbol{x}_k, n_k = 1) = \max(&w_1 \Pr^*[r_N = r^1 \mid \boldsymbol{x}_k, n_k = 1], \\ &\cdots, \\ &w_m \Pr^*[r_N = r^m \mid \boldsymbol{x}_k, n_k = 1]),\end{aligned}$$

where $\Pr^*[\boldsymbol{r}_N = \boldsymbol{r}^j]$ *is the probability that space probe can reach at j-th candidate when it chooses the fuel minimum solution. Also, careful attention should be paid to oprator* max(·)*, because it is an operator to find the maximum value of argument unlike other operator* max *to find the maximum value of the function.*

Proof. When only one engine is normal at stage k, possibility to reach j-th candidate is expressed as

$$\begin{aligned}\Pr[\boldsymbol{r}_N = \boldsymbol{r}^j \mid \boldsymbol{x}_k, n_k] &= \prod_{i=k}^{N-1} P(1 \,|\, \boldsymbol{u}_i, 1) \\ &= \prod_{i=k}^{N-1} (1-p)^{\|\boldsymbol{u}_i\|/\bar{u}} \\ &= (1-p)^{\frac{1}{\bar{u}} \sum_{i=k}^{N-1} \|\boldsymbol{u}_i\|},\end{aligned}$$

$p \in \mathbb{R}$ is the failure possibility when the norm of control input $\bar{u} \in \mathbb{R}$ is conducted. In addition, $\boldsymbol{u}_i$ is admissible control to arrive at j-th candidate.

Since failure possibility p satisfies $0 < p < 1$, possibility to reach is maximized when the fuel consumption $\sum_{i=k}^{N-1} \|\boldsymbol{u}_i\|$ is minimized. Hereafter, we express the probability to reach when a space probe select the fuel minimum solution as

$$\Pr^*[r_N = r^j \mid \boldsymbol{x}_k, n_k], \quad \forall j = 1, \cdots, m. \tag{4}$$

Consider multiple celestial bodies, it is reasonable to select the fuel minimum solution towards each candidate in case that only one engine is normal. Hence, it is optimal to choose a celestial body which has the maximum expected value of mission accomplishment. That is, the value function when only one engine is normal, is defined as

$$\begin{aligned}V_k(\boldsymbol{x}_k, n_k = 1) = \max(&w_1 \Pr^*[r_N = r^1 \mid \boldsymbol{x}_k, n_k = 1], \\ &\cdots, \\ &w_m \Pr^*[r_N = r^m \mid \boldsymbol{x}_k, n_k = 1]).\end{aligned}$$

□

Algorithm 1 Proposed Sampling-based DP Algorithm

Require: Number of random nodes at k-th stage M_k, control input $\boldsymbol{u}_k$, number of normal engines n_k, transition probability on the number of normal engines $P(n_{k+1} \mid \boldsymbol{u}_k, n_k)$

Ensure: Approximately optimal control policy while guaranteeing probabilistic completeness

1: Generate initial reference trajectories.
2: **repeat**
3: Place M_{N-1} nodes within reachable set from the node on reference trajectory at stage $N-2$ and space where the node at N-th stage (i.e. each candidate) is reachable.
4: Connect M_{N-1} new nodes with the node at N-th stage
5: **for** $n_{N-1} = 0$ to $n_{\max}$ **do**
6: Calculate $J_{N-1}(\boldsymbol{x}_{N-1}, n_{N-1})$ for M_{N-1} nodes
7: **end for**
8: **for** $k = N-2$ to 2 **do**
9: Randomly place M_k nodes within both of space reachable from the node on reference trajectory at stage $k-1$ and the space where that at stage $k+1$ is reachable.
10: Temporally connect M_k nodes with all the nodes at stage $k+1$ and derive $\boldsymbol{u}_k$ for each transition
11: **for** $n_k = 0$ to $n_{\max}$ **do**
12: Calculate $P(n_{k+1} \mid \boldsymbol{u}_k, n_k)$ and $J_k(\boldsymbol{x}_k, n_k)$
13: Choose the node at stage $k+1$ which realizes the maximum $J_k(\boldsymbol{x}_k, n_k)$, and actually connect
14: **end for**
15: **end for**
16: Connect one node at stage 1 with M_2 nodes at stage 2
17: Calculate $J_1(\boldsymbol{x}_1, n_1 = n_{\max})$ for M_2 nodes and update with the trajectory which realize the maximum value of $J_1(\boldsymbol{x}_1, n_1 = n_{\max})$
18: **until** $J_1(\boldsymbol{x}_1, n_1 = n_{\max})$ converges

4. Proposed Method

Bellman equation can be numerically solved by various kinds of algorithms. Dynamic Programming (DP) such as policy iteration or value iteration[19] has received considerable attention since the introduction of the Bellman equation. Another mainstream is a Reinforcement Learning (RL) approach[15] such as Q-Leaning or TD-Learning. Both of the two mainstreams are based on the strong theoretical background and has been applied to many real problems. However, they cannot work well for the real problem with high dimensionality, which is known as *Chaos of Dimensionality.*

In the field of robotics, sampling based method has been frequently used in order to solve the optimal control problems. In particular, Rapidly-exploring Random Tree (RRT) algorithm[20] has been applied to many real problems such as path planning of automobiles.[21] RRT-based algorithms have been well studied in the robotics community, and various modification has been conducted such as Chance Constrained-RRT (CC-RRT).[22] Although original RRT algorithm is theoretically immature, Karaman & Frazzoli (2011) proposed RRT* algorithm[23] which has a property of asymptotic optimality in addition to probabilistic complete. Recently, sampling based algorithms with strong theoretical background, have been proposed, and most of them has relatively lighter computational complexity than general algorithms of DP or RL.

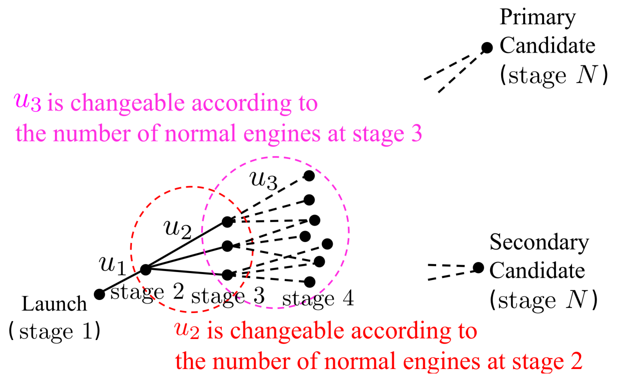

Fig. 2. Resulting trajectory branches according to the number of normal engines.

Our proposed algorithm is a sampling-based algorithm developed to intelligently identify and refine the probabilistically optimal trajectory with backups considering possible faults of engines. By applying Bellman's principle of optimality to RRT, this algorithm can search optimal solution by efficient computation. In addition, only solutions satisfying the boundary conditions can be remained through iterations because this algorithm extends trees considering the reachable sets.

In summary, our sampling-based DP method generates a sequence of controls. At each iteration, the following main steps must be performed. First, an admissible policy is found and defined as reference trajectory. Then, the admissible policy from stage k that increases the object function is searched around the reference trajectory by solving a maximization subproblem in a backward sweep. After conducting this procedure until stage 1, the reference trajectory is updated to the trajectory whose object function is maximum. The sequence above is iterated until convergence.

The following subsections provide more details on our sampling-based DP algorithm. The line number in subsection heading corresponds to that of Algorithm 1.

4.1. Generation of Initial Reference Trajectory: line 1

Initial reference trajectory is required for our proposed method. Because the optimal policy is changeable according to the number of normal engines, the reference trajectory is branching. In addition, the trajectory can also branch off at every stage according to the target celestial body. Therefore, the reference trajectory consists of at most $(m \times n_{\max})^N$ branches. In general, this number is so large that we have to efficiently sample new nodes.

4.2. Node Sampling: line 2, 9

Due to the high-dimensional and continuous state space, it is significant to efficiently sample and refine the new state. Our proposed method realizes its efficiency by introducing the concept of reachable set. This method generates new nodes at stage k within both the area where the node at stage $k+1$ on reference trajectory is reachable and reachable set from the node at stage $k-1$. That is, such control inputs $\boldsymbol{u}_{k-1}(\boldsymbol{x}_{k-1}, n_{k-1})$ and $\boldsymbol{u}_k(\boldsymbol{x}_k, n_k)$ satisfy the following inequality.

$$\boldsymbol{u}^{LB}(n_{k-1}) \leq \boldsymbol{u}_{k-1}(\boldsymbol{x}_{k-1}, n_{k-1}) \leq \boldsymbol{u}^{UB}(n_{k-1}),$$
$$\boldsymbol{u}^{LB}(n_k) \leq \boldsymbol{u}_k(\boldsymbol{x}_k, n_k) \leq \boldsymbol{u}^{UB}(n_k)$$

By sampling new nodes within the area, only the nodes which can satisfy the boundary conditions can be chosen. This property enables the space probe to prevent from continuing to consider the solutions which cannot satisfy boundary conditions of boundary value problems.

4.3. Tree Extension: line 4, 10-13

New nodes at stage k are temporally connected to all nodes at stage $k+1$. Because state vector $\boldsymbol{x}_k$ and $\boldsymbol{x}_{k+1}$ must satisfy the state equation, the control input $\boldsymbol{u}_k$ to transfer from $\boldsymbol{x}_k$ to $\boldsymbol{x}_{k+1}$ can be calculated. The failure possibility of low-thrust propulsion systems is a function of the norm of the control input $\boldsymbol{u}_k$. The transition probability $P(n_{k+1} \mid \boldsymbol{u}_k, n_k)$ is known, so one node at stage k+1 realizing the maximum value of object function can be determined. In the end, a new node at stage k is actually connected with the node at stage $k+1$ which realizes the maximum value of object function. Nodes at stage $k+1$ which do not connect with any node at stage k are removed.

4.4. Update of Reference Trajectory: line 17

Node sampling (Subsection B) and tree extension (Subsection C) are conducted from stage N to stage 1. The trajectory realizing the maximum value of object function $J_1(\boldsymbol{x}_1, n_1 = n_{\max})$ is defined as a reference trajectory. At this time, optimal policy according to the number of normal engines for every stage can be obtained (see Figure 2). Our algorithm conducts node sampling and tree extension based on the updated reference trajectory until $J_1(\boldsymbol{x}_1, n_1 = n_{\max})$ converges.

5. Theoretical Analysis

In this section, theoretical analysis on a proposed method is provided. In the sampling-based motion/path planning community, it does matter whether the algorithm has two significant properties or not: *probabilistic complete* and *asymptotic optimality*. For more detail of sampling based motion planning algorithm, see Karaman & Frazzoli (2011)[23] Our proposed method possesses the property *probabilistic complete*, but not *asymptotic optimality.* In this section, the theoretical analysis on *probabilistic complete* is described.

Definition 1. *(Probabilistic completeness). Let V_n and E_n denote vertex set and edge set after n iteration, respectively. Also, Graph G_n can be denoted as $G_n = (V_n, E_n)$, where $V \subset \mathcal{X}_{free}$, $card(V) \leq n+1$, and $E \in V \times V$. An algorithm is probabilistic complete, if, for any robustly feasible trajectory/path planning problem $(\mathcal{X}, \mathcal{X}_{init}, \mathcal{X}_{goal})$,*

$$\liminf_{n\to\infty} \Pr(\exists x_{goal} \in V_n \cup \mathcal{X}_{goal} \text{ such that } x_{init} \text{ is connected to } x_{goal} \text{ in } G_n) = 1$$

If an algorithm is probabilistically complete, and the path or trajectory planning problem is robustly feasible, the limit

$$\lim_{n\to\infty} \Pr(\exists x_{goal} \in V_n \cup \mathcal{X}_{goal} \text{ such that } x_{init} \text{ is connected to } x_{goal} \text{ in } G_n)$$

exists and is equal to one.

Theorem 3. *Our proposed method has the property of probabilistic complete.*

Proof. At stage k, proposed randomly places the next node within the reachable set from the reference node $k-1$. That is, the follows hold:

$$\Pr(\exists x_{\text{init}} \in V_n \cup \mathcal{X}_{\text{init}} \text{ such that } x_{\text{goal}} \text{ is connected to } x_{\text{init}} \text{ in } G_n) = 1,$$

where our proposed method method relies on backward sweep, so x_{init} and x_{goal} have been swapped comparing with the definition of *probabilistic complete*. The above equation is equivalent with the definition written as

$$\Pr(\exists x_{\text{goal}} \in V_n \cup \mathcal{X}_{\text{goal}} \text{ such that } x_{\text{init}} \text{ is connected to } x_{\text{goal}} \text{ in } G_n) = 1$$

□

Our proposed method does not possess the property of *asymptotic optimality*; that is, the algorithm does not guarantee to derive the exactly optimal solution. However, our algorithm is guaranteed to continue to derive solutions satisfying the boundary conditions. Furthermore, because of procedure to update the reference trajectory (subsection 4.4), the value will also be guaranteed to increase monotonically.

6. Experiment

6.1. Problem Settings

The new method for trajectory design was implemented and tested using a real deep space probe trajectory design scenario. This section shows the key results from this test. We conducted the simulation supposing that a deep space probe with two ion thrusters explores an asteroid by employing Earth Gravity Assist (EGA) one year after launched. Reference trajectory of relative vectors is assumed to be the trajectory of Earth. Simulation condition is defined as Table 1 and Table 2. All the vectors in this table are relative to the trajectory of Earth. Consider the linear time-variant system. The state equation is denoted as

$$\begin{bmatrix} \boldsymbol{r}_{k+1} \\ \boldsymbol{v}_{k+1} \end{bmatrix} = \begin{bmatrix} \Phi_{11}(t_k, t_{k+1}) & \Phi_{12}(t_k, t_{k+1}) \\ \Phi_{21}(t_k, t_{k+1}) & \Phi_{22}(t_k, t_{k+1}) \end{bmatrix} \begin{bmatrix} \boldsymbol{r}_k \\ \boldsymbol{v}_k \end{bmatrix} + \begin{bmatrix} \Phi_{12}(t_k, t_{k+1}) \\ \Phi_{22}(t_k, t_{k+1}) \end{bmatrix} \boldsymbol{u}_k, \tag{5}$$

where $\Phi_{ij}(t_k, t_{k+1}) \in \mathbb{R}^{3\times3}, i, j = 1, 2$ is the State Transition Matrix (STM) from stage k to stage $k+1$. $\boldsymbol{r}_k \in \mathbb{R}^3, \boldsymbol{v}_k \in \mathbb{R}^3$ are the position and velocity relative to Earth, respectively. $\boldsymbol{u}_k \in \mathbb{R}^3$ is the velocity increment by ion engines. Because of Equation 5, the relative position vector at stage N can be described as

$$\boldsymbol{r}_N = \Phi_{11}(t_1; t_N)\boldsymbol{r}_1 + \Phi_{12}(t_1; t_N)\boldsymbol{v}_1 + \sum_{i=1}^{N-1} \Phi_{12}(t_i; t_N)\boldsymbol{u}_i. \tag{6}$$

Deviation of Equation 6 is shown in Appendix. In this simulation, simultaneous operation of more than one engines is not allowed. Hence, transition probability of normal engines is expressed as

$$P(n_{k+1} | \boldsymbol{u}_k, n_k) = \begin{cases} (1-p)^{\frac{\|\boldsymbol{u}_k\|}{\bar{u}}} & \text{if } n_{k+1} = n_k \\ p^{\frac{\|\boldsymbol{u}_k\|}{\bar{u}}} & \text{if } n_{k+1} = n_k - 1 \\ 0 & \text{otherwise,} \end{cases}$$

where $p \in \mathbb{R}$ is the possibility of failure when $\bar{u} \in \mathbb{R}$ velocity increment is conducted.

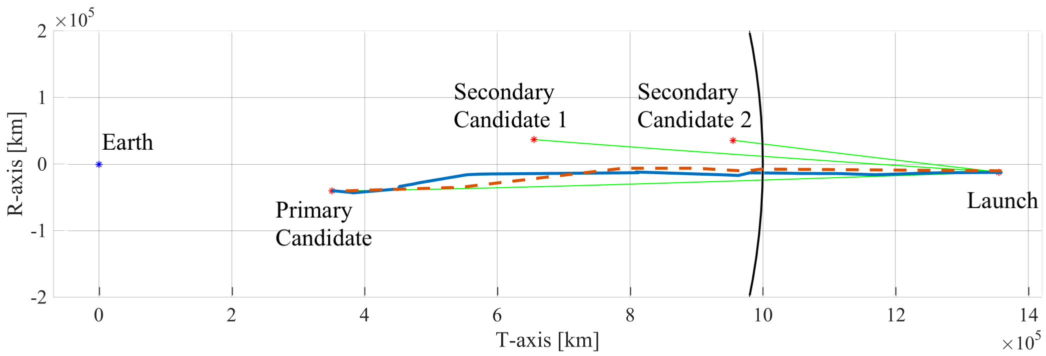

Fig. 3. Resulting trajectory of the simulation with 2% failure possibility. Observe that blue trajectory bends towards secondary candidates. Blue line and red dotted line mean the optimal trajectory in case of 2% failure possibility and 3% failure possibility, respectively.

Table 1. Simulation Condition

Mass of spacecraft [kg]	65.0
Thrust [μN]	350
Number of thrusters	2
Initial position [km] Initial velocity [km/s] Terminal velocity [km/s]	$(x, y, z) = (0, 0, 0)$ $(u, v, w) = (-0.015,\ 3.6,\ 2.8)$ (u, v, w) = (Free, Free, Free)
Orbit transfer time [day]	365
Number of segment	73
Exploration worth	Primary Candidate: 1.0 Secondary Candidate 1: 0.8 Secondary Candidate 2: 0.8

Table 2. Position of each candidate on B plane

Candidate	Position on B plane [km]
Primary Candidate	$(3.57 \times 10^5, -4.05 \times 10^4)$
Secondary Candidate 1	$(6.55 \times 10^5,\ 3.70 \times 10^4)$
Secondary Candidate 1	$(9.55 \times 10^5,\ 3.60 \times 10^4)$

Table 3. Value function at stage 1

	Failure possibility	
	2%	3%
Fuel minimum solution to P.C. targeting only P.C.	0.7784	0.6153
Fuel minimum solution to P.C. targeting all candidates	0.7961	0.6289
Optimal solution in this simulation	0.8544	0.7530

6.2. Results

Figure 3 shows the optimal trajectory planned by our proposed method, which means the trajectory on B plane. While fuel minimum trajectories to each candidate (green lines) are almost straight on B plane, the optimal trajectories in this research (blue line and red dotted line) bend towards secondary candidates. Blue line and red dotted line mean the optimal trajectories in the case of 2% and 3% possibility, respectively. Table 3 compares the value function at stage 1, $V_1(\boldsymbol{x}_1, n_1 = 2)$ with the value of the objective function realized by fuel minimum solution to the primary candidate. The value of the object function by the optimal solution is approximately 10% and 15% improved compared with that by the conventional solution in simulations with 2% and 3% failure possibility, where conventional solution means the fuel minimum solution to primary candidate aiming at only primary candidate.

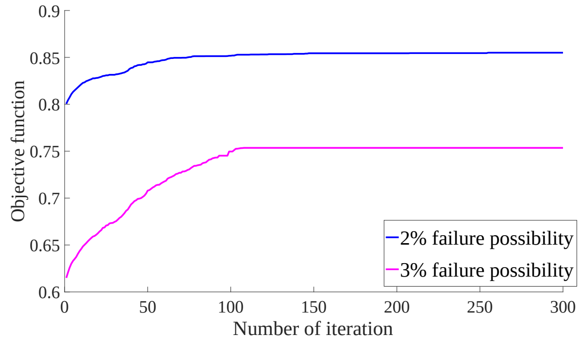

Fig. 4. Relation between the number of iteration and the maximum value of the objectiveive function, $J_1(\boldsymbol{x}_1, n_1 = 2)$.

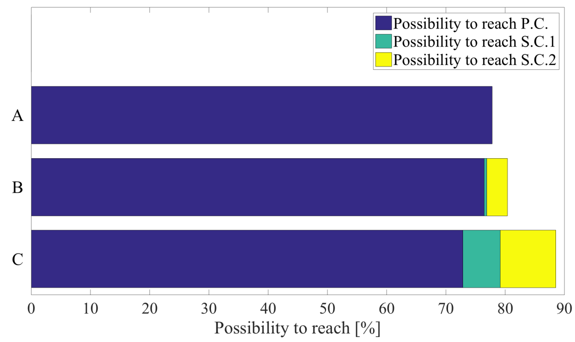

Fig. 5. Possibility to reach each candidate when defining the failure possibility as 2%. "A" indicates the fuel minimum solution to primary candidate aiming at only primary candidate, which is conventional method. "B" indicates the fuel minimum solution to primary candidate considering secondary candidates. "C" indicates that optimal solution in this simulation.

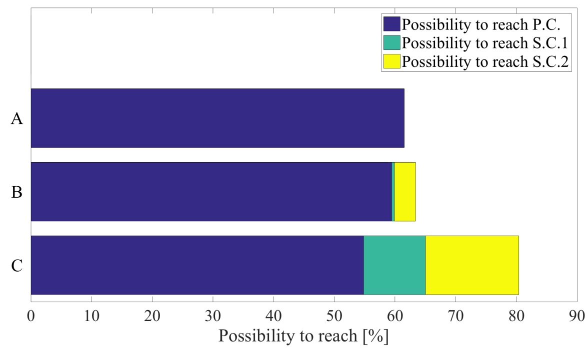

Fig. 6. Possibility to reach each candidate when defining the failure possibility as 3%. The meaning of capital letters A, B, and C is described in the caption of Figure 5.

Figure 4 indicates the relation between the number of iteration and the objective function. Observe that the value of the objective function converges after almost 100 - 170 iterations. Our proposed method does not possess the property of *asymptotic optimality*, but it can be estimated that the obtained solution is favorable.

Figure 5 shows the arrival possibility to each candidate when assuming failure possibility to be 2%. Arrival possibility to primary candidate realized by the optimal solution in this research is 5.96% lower than that by conventional solution. However, the probability that a space probe cannot reach any asteroid can be reduced by 10.7%. Figure 6 shows the arrival possibility to each candidate with 3% failure possibility. Arrival possibility to primary candidate realized by the optimal solution in this research is 6.71% lower than that by fuel minimum solution aiming at only primary candidate. However, the probability that a space probe cannot reach any asteroid can be reduced by 18.89%. The trend is marked comparing with the simulation result with 2% failure possibility.

7. Conclusion and Future Work

In this paper, we proposed a method to design trajectories robust to the possible permanent failure of low thrust engines. We formulated the finite-horizon optimal control problem with state and control constraint, which maximizes the expected scientific gain. The objective function is defined as the weight sum of probability to reach each target celestial body. This objective appears to be incompatible with conventional approach such as fuel minimum solution, but theoretical analysis proves that fuel minimum solution can be obtained as the optimal solution for the specified problem of our formulation. In order to solve this problem with a reasonable computational time, we presented a sampling based dynamic programming algorithm, specifically developed for a space probe with multiple engines considering backup trajectories. Theoretical analysis proves that this algorithm possesses a property of probabilistic completeness, which is one of the significant property for sampling based algorithm. We demonstrated the efficiency of our algorithm using real trajectory design scenario.

Future work will focus on the reduction of computational complexity of proposed method and device for coming up with an algorithm with the property of asymptotic optimality. It will be important to implement a proposed method in a path planning scenario for Mars rover or a trajectory design scenario for the spacecraft with more than two thrusters.

References

1) Jean-Francois Castet and Joseph H Saleh. Satellite and satellite subsystems reliability: Statistical data analysis and modeling. *Reliability Engineering & System Safety*, Vol. 94, No. 11, pp. 1718–1728, 2009.
2) Mark Smotherman and Kay Zemoudeh. A non-homogeneous markov model for phased-mission reliability analysis. *IEEE Transactions on Reliability*, Vol. 38, No. 5, pp. 585–590, 1989.
3) Brian C Williams, Michel D Ingham, et al. Model-based programming of intelligent embedded systems and robotic space explorers. *Proceedings of the IEEE*, Vol. 91, No. 1, pp. 212–237, 2003.
4) Joris T Olympio. Designing robust low thrust interplanetary trajectories subject to one temporary engine failure. In *Proceedings of the 20th AAS/AIAA Space Flight Meeting*, pp. 10–171, 2010.
5) Marc D Raymana, Thomas C Fraschettia, Carol A Raymonda, and Christopher T Russellb. Preparing for the dawn mission to vesta and ceres. 2005.
6) John T Betts. Survey of numerical methods for trajectory optimization. *Journal of guidance, control, and dynamics*, Vol. 21, No. 2, pp. 193–207, 1998.
7) Jon A Sims and Steve N Flanagan. Preliminary design of low-thrust interplanetary missions. In *AAS/AIAA Astrodynamics Specialist Conference, AAS Paper*, pp. 99–338, 1999.
8) V Coverstone-Carroll, JW Hartmann, et al. Optimal multi-objective low-thrust spacecraft trajectories. *Computer methods in applied mechanics and engineering*, Vol. 186, No. 2, pp. 387–402, 2000.
9) Seungwon Lee, Paul von Allmen, et al. Comparison of multi-objective genetic algorithms in optimizing q-law low-thrust orbit transfers. In *GECCO Conference Late-breaking Paper, Washington, DC*, 2005.
10) Randal W Beard, Timothy W McLain, et al. Coordinated target assignment and intercept for unmanned air vehicles. *Robotics and Automation, IEEE Transactions on*, Vol. 18, No. 6, pp. 911–922, 2002.
11) Ron Alterovitz, Thierry Siméon, et al. The stochastic motion roadmap: A sampling framework for planning with markov motion uncertainty. In *Robotics: Science and Systems*, Vol. 3, pp. 233–241. Citeseer, 2007.
12) TBMJ Ouarda and JW Labadie. Chance-constrained optimal control for multireservoir system optimization and risk analysis. *Stochastic environmental research and risk assessment*, Vol. 15, No. 3, pp. 185–204, 2001.
13) Ryu Funase, Hiroyuki Koizumi, et al. 50kg-class deep space exploration technology demonstration micro-spacecraft procyon. 2014.
14) Akifumi Wachi, Naoya Ozaki, and Shinichi Nakasuka. Fault-tolerant low-thrust trajectory design with backups for multiple targets. *IFAC-PapersOnLine*, Vol. 49, No. 17, pp. 391–396, 2016.
15) Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, Vol. 1. MIT press Cambridge, 1998.
16) Dimitri P Bertsekas and John N Tsitsiklis. Neuro-dynamic programming: an overview. In *Decision and Control, 1995., Proceedings of the 34th IEEE Conference on*, Vol. 1, pp. 560–564. IEEE, 1995.
17) Richard Bellman. Dynamic programming, 1957.
18) Richard Ernest Bellman and Stuart E Dreyfus. Applied dynamic programming. 1962.
19) Ronald A Howard. Dynamic programming and markov processes.. 1960.
20) Steven M LaValle. Rapidly-exploring random trees a new tool for path planning. 1998.
21) Yoshiaki Kuwata, Sertac Karaman, et al. Real-time motion planning with applications to autonomous urban driving. *Control Systems Technology, IEEE Transactions on*, Vol. 17, No. 5, pp. 1105–1118, 2009.
22) Brandon Luders, Mangal Kothari, et al. Chance constrained rrt for probabilistic robustness to environmental uncertainty. In *AIAA guidance, navigation, and control conference (GNC), Toronto, Canada*, 2010.
23) Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, Vol. 30, No. 7, pp. 846–894, 2011.

Appendix 1 (Proof of Equation 6)

In this appendix, the following equation (Equation 6) is proved.

$$\boldsymbol{r}_N = \Phi_{11}(t_1; t_N)\boldsymbol{r}_1 + \Phi_{12}(t_1; t_N)\boldsymbol{v}_1 + \sum_{i=1}^{N-1} \Phi_{12}(t_i; t_N)\boldsymbol{u}_i.$$

Proof. In Equation 6, reference of the trajectory is assumed to be Earth. Hereafter, we denote the State Transition Matrix (STM) from time t_k to t_l as

$$\Phi^{l,k}(t_k; t_l) = \begin{bmatrix} \Phi_{11}^{l,k} & \Phi_{12}^{l,k} \\ \Phi_{21}^{l,k} & \Phi_{22}^{l,k} \end{bmatrix}.$$

By using STM, the following relation between state vector at time t_k and that at time t_l is satisfied.

$$\begin{bmatrix} \boldsymbol{r}(t_l) \\ \boldsymbol{v}(t_l) \end{bmatrix} = \Phi(t_k; t_l) \begin{bmatrix} \boldsymbol{r}(t_k) \\ \boldsymbol{v}(t_k) \end{bmatrix},$$

where $\boldsymbol{r}(t_k)$) and $\boldsymbol{v}(t_k)$ are relative vector against reference trajectory. Let t_N denote the terminal time: The following relation between time t_N and t_{N-1} is satisfied.

$$\begin{bmatrix} \boldsymbol{r}(t_N) \\ \boldsymbol{v}(t_N) \end{bmatrix} = \Phi(t_{N-1}; t_N) \begin{bmatrix} \boldsymbol{r}(t_{N-1}) \\ \boldsymbol{v}(t_{N-1}) \end{bmatrix}. \tag{7}$$

$\boldsymbol{v}(t_{N-1})$ can be divided into the ballistic velocity and velocity increment at time t_{N-1} and written as

$$\boldsymbol{v}(t_{N-1}) = \boldsymbol{v}^B(t_{N-1}) + \Delta v(t_{N-1}).$$

Therefore, Equation 7 can be transformed as follows:

$$\begin{bmatrix} \boldsymbol{r}(t_N) \\ \boldsymbol{v}(t_N) \end{bmatrix} = \Phi(t_{N-1}; t_N) \begin{bmatrix} \boldsymbol{r}(t_{N-1}) \\ \boldsymbol{v}^B(t_{N-1}) \end{bmatrix} + \Phi(t_{N-1}; t_N) \begin{bmatrix} 0 \\ \Delta v(t_{N-1}) \end{bmatrix}$$

The same relation as Equation 7 is satisfied between time t_{N-1} and time t_{N-1}, so the follows holds:

$$\begin{bmatrix} \boldsymbol{r}(t_{N-1}) \\ \boldsymbol{v}^B(t_{N-1}) \end{bmatrix} = \Phi(t_{N-2}; t_{N-1}) \begin{bmatrix} \boldsymbol{r}(t_{N-2}) \\ \boldsymbol{v}(t_{N-2}) \end{bmatrix}$$

By employing the above equation, state vector at time N, $[\boldsymbol{r}(t_N), \boldsymbol{v}^B(t_N)]^\top$ can be expressed as:

$$\begin{aligned} \begin{bmatrix} \boldsymbol{r}(t_N) \\ \boldsymbol{v}(t_N) \end{bmatrix} &= \Phi(t_{N-1}; t_N)\Phi(t_{N-2}; t_{N-1}) \begin{bmatrix} \boldsymbol{r}(t_{N-2}) \\ \boldsymbol{v}(t_{N-2}) \end{bmatrix} \\ &\quad + \Phi(t_{N-1}; t_N) \begin{bmatrix} 0 \\ \Delta v(t_{N-1}) \end{bmatrix} \end{aligned} \tag{8}$$

Because STM has a property of

$$\Phi(t_{N-1}; t_N)\Phi(t_{N-2}; t_{N-1}) = \Phi(t_{N-2}; t_N),$$

Equation 8 can be simply denoted as:

$$\begin{bmatrix} \boldsymbol{r}(t_N) \\ \boldsymbol{v}(t_N) \end{bmatrix} = \Phi(t_{N-2}; t_N) \begin{bmatrix} \boldsymbol{r}(t_{N-2}) \\ \boldsymbol{v}(t_{N-2}) \end{bmatrix} + \Phi(t_{N-1}; t_N) \begin{bmatrix} 0 \\ \Delta v(t_{N-1}) \end{bmatrix}$$

By repeating this procedure down to time t_1, we can obtain the following equation:

$$\begin{bmatrix} \boldsymbol{r}(t_N) \\ \boldsymbol{v}(t_N) \end{bmatrix} = \Phi(t_1; t_N) \begin{bmatrix} \boldsymbol{r}(t_1) \\ \boldsymbol{v}^B(t_1) \end{bmatrix} + \sum_{i=1}^{N-1} \Phi(t_i; t_N) \begin{bmatrix} 0 \\ \Delta v(t_i) \end{bmatrix}$$

In the end, we can finally obtain Equation 6, which orbit equation used in the experiment:

$$\boldsymbol{r}(t_N) = \Phi_{11}(t_1; t_N)\boldsymbol{r}(t_1) + \Phi_{12}(t_1; t_N)\boldsymbol{v}^B(t_1) + \sum_{i=1}^{N-1} \Phi_{12}(t_i; t_N)\Delta v(t_i)$$

$$\boldsymbol{r}_N = \Phi_{11}(t_1; t_N)\boldsymbol{r}_1 + \Phi_{12}(t_1; t_N)\boldsymbol{v}_1 + \sum_{i=1}^{N-1} \Phi_{12}(t_i; t_N)\boldsymbol{u}_i.$$

Transformation from first line to second is derived by the problem setting that $\boldsymbol{u}_i$ is equivalent with $\Delta v(t_i)$. □