DOI: ADD DOINUMBER HERE

Chance-Constrained Optimal Altitude Control of a Rocket

Thomas Lew, Fabian Lyck and GianAndrea Müller
Department of Mechanical and Process Engineering, ETH Zürich, Switzerland
{lewt,flyck,muellegi}@student.ethz.ch

Abstract

Motivated by the problem of launching a rocket to reach a target apogee altitude, this paper presents an algorithm solving a chance-constrained infinite horizon optimal control problem. Using actuated airbrakes to control longitudinal drag forces, it minimizes the apogee altitude error while respecting input rate chance constraints. The effects of parameters uncertainty and disturbances are estimated using Monte Carlo simulations and incorporated in the problem. The solution is stored in memory as a look-up control table on a computationally limited rocket, which is successfully flown with an apogee altitude error of 1% for a target altitude of 800m.

Nomenclature

Notations

$\mathcal{N}(\mu, \Sigma)$	Gaussian distribution of mean μ and variance Σ
$\wedge$	conjuction (logical AND)
$\vee$	disjunction (logical OR)

Variables

$\mathbf{x}$	rocket state
z, v	altitude and vertical velocity
u	normalized airbrakes position
$\mathbf{w}_k$	Gaussian-distributed disturbance
$\mu_w(\cdot)$	mean of disturbance
$\Sigma_w(\cdot)$	variance of disturbance
λ	input rate constraint dual variable
$\mu(\cdot)$	feedback control policy
$J(\cdot)$	optimal cost

Functions

$f(\cdot)$	nominal dynamics
$f_0(\mathbf{x}_{pad}, \zeta_j)$	state at motor burnout from $\mathbf{x}_{pad}$ for MC parameter ζ_j
$g(\cdot)$	step cost
$g_{apg}(\cdot)$	apogee cost
$\bar{I}_u(\cdot), I_u(\cdot)$	exact and smooth input rate constraint indicator functions
$I_{crit}(\cdot)$	critical region indicator function
$r_u^\mu(\mathbf{x})$	risk to violate the input rate constraint from $\mathbf{x}$ using μ
$r_{crit}^\mu(\mathbf{x})$	risk to enter $\mathcal{X}_{crit}$ from $\mathbf{x}$ using μ

Abbreviations

MC	Monte Carlo
DP	Dynamic Programming
VI	Value Iteration
DoF	degree of freedom
CoM, CoP	center of mass, center of pressure

Parameters and Constants

$\mathbf{x}_{pad}, \mathbf{x}_{goal}$	initial and goal states
h_{goal}	target altitude
Δh_{goal}	max. apogee height deviation
Δt	dynamics discretization time
M_{id}	number of MC simulations
M_0	number of burnout positions
Δ_u	input rate constraint probability threshold
b_u	maximum actuation difference
$\dot{u}_{max}$	maximum airbrakes velocity
$t_{min}(\mathbf{x})$	min. time to reach $\mathbf{x}$ from $\mathbf{x}_{pad}$
N_m	time to apogee for MC trajectory
J_i	number of MC samples for node $\mathbf{x}_i$
i_{max}	maximum number of iterations of the bisection method

Sets

$\mathcal{X}$	controllable region
$\mathcal{X}_{apg}$	apogee region
$\mathcal{X}_{crit}$	critical region
$\mathcal{X}_0$	burnout region
$\mathcal{X}_{end}$	end region
$\mathcal{U}$	feasible inputs set
$\gamma_{min}(v)$, $\gamma_{max}(v)$	limiting trajectories for full/zero airbrakes deployment

1. Introduction

Controlling a nonlinear system with uncertain parameters and external disturbances is a challenging task, especially for aerospace applications which require high success guarantees. In this work, we consider the design of a guidance and control algorithm for a student-built rocket for the Spaceport America Cup[1], the world's largest intercollegiate rocket engineering competition. The event consists of the launch of a rocket to precisely reach a predefined height at apogee. By including actuated airbrakes on our system, our team is able to control longitudinal drag forces to adjust the rocket's ascent speed and reach the target altitude. The design of such an algorithm is particularly challenging due to the inherent nonlinear and uncertain dynamics (e.g., drag forces), external disturbances (e.g., wind) and physical constraints, such as actuator delays, limited memory and computing capabilities.

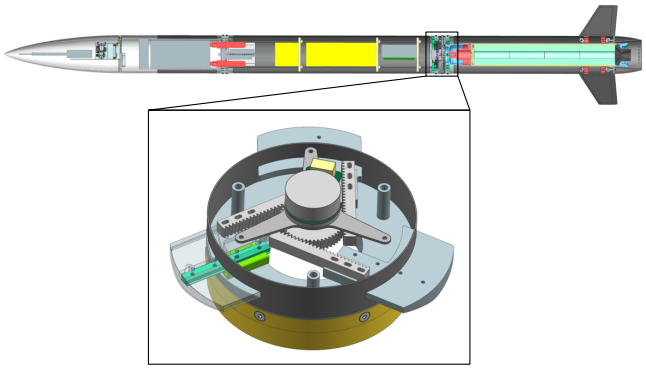


Figure 1: Rocket with airbrakes: surfaces controlled by a servomotor which can increase the longitudinal drag forces. The airbrakes are located behind the center of pressure to guarantee stability.

In presence of external random disturbances and model mismatch, the problem of reaching a target apogee while satisfying constraints can be expressed as a chance-constrained optimal control problem. Instead of using a worst-case analysis as in robust model predictive control methods [3], which assume bounded disturbances and model mismatch, chance constraints ensure the satisfaction of constraints with a predefined probability. Chance constraints are present in many applications [1, 5, 6, 12], including aerospace applications [8,9,15,20,26]. Taking into account chance constraints generally requires to reformulate them as deterministic constraints, so the resulting nonlinear optimization problem can be solved using an off-the-shelf nonlinear programming solver. In [8] and [26] for instance, Monte Carlo sampling of the random parameters is used and the chance constraints are approximated using the kernel density estimation technique and a split-Bernstein approach, respectively. In both case, computationally expensive Monte Carlo (MC) sampling prohibits the computation of trajectories in real time.

Another approach to solve optimal control problems on discretized state-input spaces is Dynamic Programming (DP). By defining independent joint constraints and using Boole's inequality, [20] presented a chance-constrained DP algorithm to solve such a problem for a system subject to additive Gaussian-distributed disturbances of known mean and variance. In practical applications, the characteristics of those Gaussian disturbances are unknown and the different sources of uncertainty and modeling errors which they represent can be difficult to identify. For instance, due to nonlinear drag forces which depend on the velocity of the rocket and on uncertain parameters, typical assumptions on the disturbances (e.g. Gaussian additive noise of known constant variance) do not capture the real behavior of the system.

Also, as the airbrakes actuator delays are comparable to control update rates, it is important to include input rate constraints in the problem formulation to limit the maximum rate of change of extension of the airbrakes. In general, solving optimal control problems can result in bang-bang type controllers, which would not capture the true behavior of this system. However, taking into account such constraints in DP formulations is a challenge, as traditional approaches consist of extending the state of the rocket with previous control inputs, resulting in longer computational time and requiring more on-board memory.

Furthermore, certifying guidance and control algorithms for aerospace applications and their implementation on computationally limited hardware constitute challenges in themselves. Successful optimization-based algorithms for trajectory optimization and control such as model predictive control [21] require considerable on-board computing capabilities and bring issues such as possible loss of feasibility. For this reason, a

[1]Spaceport America Cup: `https://www.spaceportamericacup.com/`

reference trajectory is typically computed offline and tracked during the flight using a feedback controller. If the initial conditions are uncertain, several reference trajectories can be generated and saved in memory on the system [22]. On the other hand, it is possible to use DP to directly compute the optimal control feedback policy on a discretized state-input space. However, this would require considerable on-board memory to store the solution and as the final time to reach apogee is unknown, work such as [20] is not directly applicable. This motivates the development of a new approach which is simple, robust, efficient and ensures the success of the flight.

The contribution of this paper is a novel guidance and control algorithm consisting of three steps. First, we use a reduced-order model of the system and incorporate the effects of external disturbances, model missmatch and parameter uncertainty as Gaussian-distributed disturbances which mean and variance are identified by running Monte Carlo simulations using a high-fidelity model of the rocket and of its environment for a range of parameters, such as aerodynamic coefficients, motor efficiencies and wind. Second, since the final time at apogee is unknown, we define the control problem as an infinite horizon chance-constrained optimal control problem. To include input rate constraints, we use a novel formulation which does not require the increase of the number of decision variables and could be used for DP and Reinforcement Learning. This problem is solved offline to generate a look-up control table which is saved in memory on the rocket using an optimized discretization scheme. Finally, during the flight, a look-up table storing the solution of the problem is accessed and control inputs are used in real time. Importantly, our algorithm provides an estimate of the probability of success of the flight, which is critical since the competition allows for a single attempt.

This paper is organized as follows. In Section 2, we formulate the chance-constrained optimal control problem, present the disturbance identification method and our novel formulation of input rate constraints. In Section 3, we reformulate the original problem using dual programming, Boole's inequality and Bellman's Principle of Optimality and present an algorithm to solve the problem. In Section 4.1, we discuss implementation details as well as convergence and optimality guarantees of the algorithm. The method presented in this paper is verified on a rocket built by our team[2] in a test flight and the results are presented in Section 4. We conclude in Section 5 and provide details about the high fidelity simulator in Appendix A.

2. Problem Formulation

The goal of the competition consists of launching a rocket from an initial state on the launchpad $\mathbf{x}_{pad}$ to precisely reach an altitude at apogee h_{goal}. The flight consists of two phases. First, the rocket flies until motor burnout, during which no control is allowed. Then, the rocket is controlled using actuated airbrakes. Since the conditions of the flight are known in advance, it is possible to compute the solution to the problem offline by simplifying the dynamics to obtain a tractable statespace which can be discretized.

2.1 Rocket Dynamics

The dynamics of the rocket can be described using a 13-dimensional state describing the 6 degrees of freedom (DoF) of the system, as described in the appendix in Equation (23). To reach the target apogee, the drag forces can be controlled by adjusting the extension of the airbrakes u. To reduce the number of state variables and since the airbrakes position u only influences the longitudinal component of the trajectory, the dynamics are simplified by removing 2 planar and 3 rotational DoF using the following assumptions:

- Pitch is a fixed function of altitude. First, a reference flight is simulated and its pitch values are stored as a function of the altitude to remove this variable.
- Only drag and gravity are the forces taken into account, with drag forces acting along the longitudinal axis of the rocket.
- No Torques, removing rotations from the DoF. Assuming the rocket to be symmetrical along its roll axis and since the drag acts directly along the longitudinal axis, no torque is induced on the rocket.
- Lateral coordinates (x, y) do not influence the final apogee altitude and are not considered.

Therefore, we consider the following 2-dimensional nonlinear stochastic discrete-time system

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, u_k) + \mathbf{w}_k(\mathbf{x}_k) \tag{1a}$$

$$\mathbf{w}_k(\mathbf{x}_k) \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{w}}(\mathbf{x}_k), \boldsymbol{\Sigma}_{\mathbf{w}}(\mathbf{x}_k)), \tag{1b}$$

[2] ARIS: Swiss Space Initiative `http://aris-space.ch/`

where $f(\cdot)$ denotes the nominal model of the dynamics, $\mathbf{x}_k = [z_k, v_k] \in \mathbb{R}^2$ denotes the rocket state with z_k the altitude and v_k the longitudinal velocity, and the control input $u_k \in \mathbb{R}$ corresponds to the airbrakes position. $\mathbf{w}_k \in \mathbb{R}^2$ corresponds to the disturbances, capturing both external disturbances and model mismatch between $f(\cdot)$ and the 6 DoF nonlinear dynamics. The dependency of $\boldsymbol{\mu}_\mathbf{w}(\cdot)$ and $\boldsymbol{\Sigma}_\mathbf{w}(\cdot)$ on the state $\mathbf{x}_k$ is crucial, since drag forces vary as a function of the speed and the wind speeds vary as a function of altitude.

2.2 Model Mismatch and Disturbances Identification using Monte Carlo Simulations

To identify $\boldsymbol{\mu}_\mathbf{w}(\cdot)$ and $\boldsymbol{\Sigma}_\mathbf{w}(\cdot)$, M_{id} MC simulations are run using a high fidelity model of the rocket with varying parameters $\{\boldsymbol{\zeta}^m\}_{m=1}^{M_{id}}$ including motor efficiency, drag coefficients, launch rail headings and wind. To ensure that the statespace is uniformly visited and improve the estimates, the simulations are run using 10 reference trajectories uniformly distributed in the statespace which are tracked with a proportional controller. This defines a dataset of transitions $\left\{\left(\mathbf{x}_k^m, u_k^m, \mathbf{x}_{k+1}^m\right)\right\}_{k=0\ldots N_m}^{m=1\ldots M_{id}}$, used to infer $\boldsymbol{\mu}_\mathbf{w}(\cdot)$ and $\boldsymbol{\Sigma}_\mathbf{w}(\cdot)$ as follows.

First, the statespace is discretized into 201 different altitude and speed values using a rectangular discretization grid, which edges are denoted as $\mathbf{x}^i$. For each transition $\left(\mathbf{x}_k^m, u_k^m, \mathbf{x}_{k+1}^m\right)$ in the dataset, a disturbance sample $\mathbf{w}^j$ for the closest node of the grid $\mathbf{x}^i := \text{argmax}(||\mathbf{x}_k^m - \mathbf{x}^i||_2)$ is computed as

$$\mathbf{w}^j(\mathbf{x}^i) = \mathbf{x}_{k+1}^m - f(\mathbf{x}_k^m, u_k^m, \boldsymbol{\zeta}^m). \tag{2}$$

For each node $\mathbf{x}^i$ with J_i samples, the local mean and variance are then computed as

$$\boldsymbol{\mu}_\mathbf{w}^i = \frac{1}{J_i}\sum_{j=1}^{J_i} \mathbf{w}^j(\mathbf{x}^i), \qquad \boldsymbol{\Sigma}_\mathbf{w}^i = \frac{1}{J_i - 1}\sum_{j=1}^{J_i} (\mathbf{w}^j(\mathbf{x}^i) - \boldsymbol{\mu}_\mathbf{w}^i)^T (\mathbf{w}^j(\mathbf{x}^i) - \boldsymbol{\mu}_\mathbf{w}^i). \tag{3}$$

Finally, $\boldsymbol{\mu}_\mathbf{w}(\cdot)$ and $\boldsymbol{\Sigma}_\mathbf{w}(\cdot)$ are computed by fitting 3rd order polynomial surfaces through $\left\{\boldsymbol{\Sigma}_\mathbf{w}^i\right\}$ and $\left\{\boldsymbol{\mu}_\mathbf{w}^i\right\}$. This method is arbitrary but enables the user to choose a parameterization for $\boldsymbol{\mu}_\mathbf{w}(\cdot)$ and $\boldsymbol{\Sigma}_\mathbf{w}(\cdot)$, which we choose as a 3rd polynomial surface since drag forces are approximately proportional to the square of the apparent velocity, defined in the appendix. As mentioned in the introduction, this is equivalent to defining a Gaussian process for $\mathbf{w}_k(\mathbf{x})$ with mean $\boldsymbol{\mu}_\mathbf{w}(\mathbf{x})$ and kernel $\boldsymbol{\Sigma}_\mathbf{w}(\mathbf{x})$ which captures prior knowledge about the system coming from Monte Carlo simulations. More details are presented in Section A.4.

2.3 Statespace Sets Definition

The goal of reaching a target altitude at apogee is not achievable for any arbitrary state $\mathbf{x} \in \mathbb{R}^2$. To allow a simple formulation of the optimal control problem, the statespace is decomposed into different regions according to the deterministic nominal dynamics $f(\cdot)$. Since the drag force is max/minimized by fully deploying/retracting the airbrakes, we define two boundary trajectories $\gamma_{min}(v)$ and $\gamma_{max}(v)$ which are used to bound the set of states $\mathcal{X}$ from which $\mathbf{x}_{goal}$ is reachable. For instance, a state $\mathbf{x} \in \{v > 0,\ h > \gamma_{max}(v)\}$ would always result in overshoot of the apogee altitude despite full airbrakes deployment. $\gamma_{min}(v)$ and $\gamma_{max}(v)$ are computed by back-propagating $f(\cdot)$ in time from the apogee altitude $h_{goal} \pm \Delta h_{goal}$, assuming full and zero deployment of the airbrakes and $\Delta h_{goal} = 5\% \cdot h_{goal}$. Note that this method provides only an approximation of the controllable region $\mathcal{X}$, since disturbances have infinite support and therefore, a disturbance to reach $\mathbf{x}_{goal}$ exists for any state $\mathbf{x}$. Using these definitions, the statespace is decomposed into a controllable region $\mathcal{X}$, an apogee region $\mathcal{X}_{apg}$, a critical region $\mathcal{X}_{crit}$, a burnout region $\mathcal{X}_0$ and an end region $\mathcal{X}_{end}$, defined as

$$\mathcal{X} = \left\{\mathbf{x} \in \mathbb{R}^2 \mid v > 0,\ \gamma_{min}(v) < h < \gamma_{max}(v)\right\} \tag{4a}$$

$$\mathcal{X}_{apg} = \left\{\mathbf{x} \in \mathbb{R}^2 \mid v \leq 0,\ \gamma_{min}(v) < h < \gamma_{max}(v)\right\} \tag{4b}$$

$$\mathcal{X}_{crit} = \mathbb{R}^2 \setminus \{\mathcal{X} \cup \mathcal{X}_{apg}\} \tag{4c}$$

$$\mathcal{X}_0 = \left\{\mathbf{x} \in \mathbb{R}^2 \mid \mathbf{x} = f_0(\mathbf{x}_{pad}, \boldsymbol{\zeta}_j),\ j{=}1\ldots M_0\right\} \tag{4d}$$

$$\mathcal{X}_{end} = \emptyset, \tag{4e}$$

where $f_0(\mathbf{x}_{pad}, \boldsymbol{\zeta}_j)$ denotes the function simulating the launch of the rocket from $\mathbf{x}_{pad}$ up to motor burnout for a set of M_0 parameters $\boldsymbol{\zeta}_j \in \mathbb{R}^p$. Each $\mathbf{x}_0 \in \mathcal{X}_0$ corresponds to a single MC simulation from $\mathbf{x}_{pad}$, including the motor and with retracted airbrakes. $\mathcal{X}_{end}$ is defined to ensure a bounded cost, as shown in Section 3.5. Any state $\mathbf{x}_k \in \mathcal{X}$ follows the dynamics in Equation (1) whereas any state $\mathbf{x}_N \in (\mathcal{X}_{apg} \cup \mathcal{X}_{\text{crit}})$ transitions with probability 1 to $\mathcal{X}_{end}$, where it remains $\forall k \geq N + 1$.

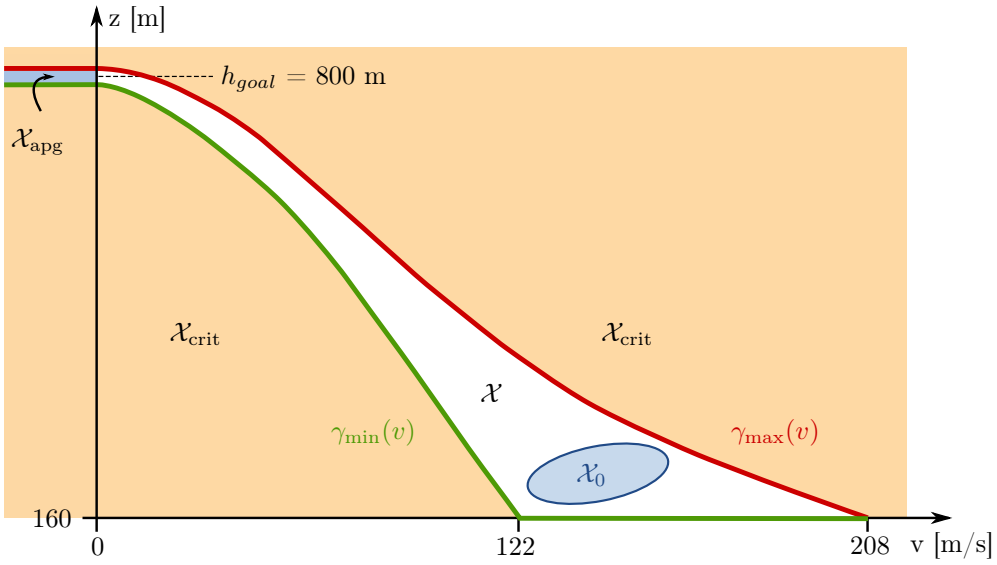


Figure 2: Statespace definition. $\gamma_{max}(v)$ and $\gamma_{min}(v)$ bound the set of states $\mathbf{x} \in \mathcal{X}$ from which the apogee target state $\mathbf{x}_{goal} = [h_{goal}; 0]$ is reachable.

2.4 Input Rate Constraints

On the rocket, the airbrakes are actuated using a servomotor which induces an actuation delay to match the actuation signal u_k. To take this delay into account, we limit the rate of change of the airbrakes position by enforcing an input rate constraint.

A straightforward approach consists of extending the state $\mathbf{x}_k$ as $\tilde{\mathbf{x}}_k{=}[\mathbf{x}_k, u_{k-1}]^T$, with u_{k-1} the previous input. Then, it is either possible to plan with the additional input rate constraint $u_k \in \{u_k \mid |u_k{-}u_{k-1}|{\leq}b_u\}$, with b_u the maximum input difference from k to $k{+}1$, or to penalize the actuation difference by adding a term to the original cost, such as $(u_k{-}u_{k-1})^2$. Both methods have two drawbacks:

1. $\tilde{\mathbf{x}}_k$ has $dim(u_k)$ more dimension(s) than $\mathbf{x}_k$, requiring more on-board memory to store the solution and longer offline computation time to compute it.
2. An estimate of u_{k-1}, computed on-board, is required.

To alleviate these problems, we use a novel formulation which consists of constraining the difference in actuation forward in time. The input rate constraint function for a feedback control policy $\mu(\cdot)$ is defined as

$$g_u(\mathbf{x}, \mu, \mathbf{w}) := (\mu\left(f(\mathbf{x}, \mu(\mathbf{x})){+}\mathbf{w}\right) - \mu(\mathbf{x}))^2 - b_u^2. \tag{5}$$

Due to the dependence on $\mathbf{w}_k$, the input rate constraints are formulated as chance constraints at probability level Δ_u as

$$\Pr\left\{g_u(\mathbf{x}_k, \mu, \mathbf{w}_k) \leq 0 \ \forall k \geq 0\right\} \geq 1 - \Delta_u. \tag{6}$$

Note that these constraints neglect the fact that the airbrakes are initially retracted until burnout ($u_0 = 0$). Given a maximum airbrakes velocity $\dot{u}_{max}$, it is possible to compute the maximum airbrakes position that can be reached t seconds after burnout. After computing the minimum time $t_{min}(\mathbf{x}_k)$ to reach each state $\mathbf{x}_k$, we enforce additional input rate constraints for the states close to burnout as

$$\begin{aligned} \mu(\mathbf{x}_k) \in \mathcal{U}(\mathbf{x}_k) &= \left[0, \min\left(\dot{u}_{max} t_{min}(\mathbf{x}_k), 1\right)\right], \\ \text{with} \quad t_{min}(\mathbf{x}_k) &= \min_{\mathbf{x}_0 \in \mathcal{X}_0} \left(t \geq 0 | \mathbf{x}_k = f_t(\mathbf{x}_0, 0)\right), \end{aligned} \tag{7}$$

where $f_t(\mathbf{x}_0, 0)$ denotes the state reached from $\mathbf{x}_0$ at time t using $u(t) = 0 \ \forall t \geq 0$. This constraint ensures that for any state $\mathbf{x}_k$ close to burnout (states for which $\dot{u}_{max} t_{min}(\mathbf{x}_k) < 1$), the control inputs are limited by the maximum airbrakes position that can be reached when starting from the initial state at burnout $\mathbf{x}_0 \in \mathcal{X}_0$ which is the closest in time to the state $\mathbf{x}_k$. Finally, for states $\mathbf{x} \in \mathcal{X}_{apg}$, we enforce $\mu(\mathbf{x}) = 1$ if $h > h_{goal}$ and $\mu(\mathbf{x}) = 0$ otherwise. As b_u depends on the discretization time Δt used for the discrete time dynamics in Equation (1), it is computed as $b_u = \dot{u}_{max} \Delta t$.

2.5 Infinite Horizon Stochastic Optimal Control Problem

Since the time to reach apogee is unknown, we state the problem as an infinite horizon optimal control problem. This will yield a time-invariant state feedback policy $\mu(\mathbf{x})$ which, for a discretized statespace, can be computed offline and stored as a look-up table on the rocket. Since the goal of the competition is to reach a precise target altitude at apogee, we define the step cost $g(\cdot)$ as

$$g(\mathbf{x}_k, \mu, \mathbf{w}_k) := \begin{cases} 0 & \text{if } \mathbf{x}_k \in \mathcal{X} \cup \mathcal{X}_{end} \\ \alpha_h (z_k - h_{goal})^2 & \text{if } \mathbf{x}_k \in \mathcal{X}_{apg} \\ 1 & \text{if } \mathbf{x}_k \in \mathcal{X}_{\text{crit}} \end{cases} \tag{8}$$

where $\alpha_h := 1/\Delta h_{goal}^2$ to enforce the continuity of $g(\cdot)$ and improve the numerical properties of our algorithm. The chance-constrained infinite horizon optimal control problem can thus be stated as

Problem 1. Chance Constrained Altitude Control Problem

$$\min_{\mu} \quad \mathbb{E}\left\{\sum_{k=0}^{\infty} g(\mathbf{x}_k, \mu, \mathbf{w}_k)\right\} \tag{9a}$$

$$\text{s.t.} \quad \Pr\left\{g_u(\mathbf{x}_k, \mu, \mathbf{w}_k) \leq 0 \ \forall k \geq 0\right\} \geq 1 - \Delta_u \tag{9b}$$

$$\mathbf{x}_k, \mathbf{w}_k \text{ satisfy (1) } \forall k \geq 0 \tag{9c}$$

$$\mu(\mathbf{x}_k) \in \mathcal{U}(\mathbf{x}_k) \ \forall k \geq 0 \tag{9d}$$

$$\mathbf{x}_0 \in \mathcal{X}_0. \tag{9e}$$

3. Risk Minimization with Input Rate Chance Constraints

In this section, we reformulate Problem 1 using Boole's Inequality and dual programming. Then, we leverage Bellman's Principle of Optimality [4] and propose an algorithm to solve the chance-constrained problem. Our approach is inspired from and extends [20] for infinite horizon chance-constrained optimal control problems.

3.1 Conservative Reformulation using Boole's Inequality

The joint chance constraint in Equation (6) can be rewritten using Boole's inequality as

$$\Pr\left\{\bigwedge_{k=0}^{\infty} g_u(\mathbf{x}_k, \mu, \mathbf{w}_k) \leq 0\right\} = 1 - \Pr\left\{\bigvee_{k=0}^{\infty} g_u(\mathbf{x}_k, \mu, \mathbf{w}_k) > 0\right\} \geq 1 - \sum_{k=0}^{\infty} \Pr\left\{g_u(\mathbf{x}_k, \mu, \mathbf{w}_k) > 0\right\}. \tag{10}$$

We then define the following exact and smooth indicator functions

$$\bar{I}_u(\mathbf{x}, \mu, \mathbf{w}) := \begin{cases} 1 & \text{if } g_u(\mathbf{x}, \mu, \mathbf{w}) > 0 \\ 0 & \text{otherwise} \end{cases}, \quad I_u(\mathbf{x}, \mu, \mathbf{w}) := \left(\frac{\mu\left(f(\mathbf{x}, \mu(\mathbf{x})) + \mathbf{w}\right) - \mu(\mathbf{x})}{b_u}\right)^2. \tag{11}$$

This allows to express each individual input rate chance constraint as a function of the expected value of $\bar{I}_u(\cdot)$, since

$$\Pr\left\{g_u(\mathbf{x}_k, \mu, \mathbf{w}_k) > 0\right\} = \mathbb{E}\left\{\bar{I}_u(\mathbf{x}_k, \mu, \mathbf{w}_k)\right\}. \tag{12}$$

Then, we rewrite the sum of probabilities in Equation (10) as

$$\sum_{k=0}^{\infty} \Pr\left\{g_u(\mathbf{x}_k, \mu, \mathbf{w}_k) > 0)\right\} = \sum_{k=0}^{\infty} \mathbb{E}\{\bar{I}_u(\mathbf{x}_k, \mu, \mathbf{w}_k)\} = \mathbb{E}\left\{\sum_{k=0}^{\infty} \bar{I}_u(\mathbf{x}_k, \mu, \mathbf{w}_k)\right\}. \tag{13}$$

Using Equations (10) and (13), a conservative reformulation of the input rate constraint (9b) is

$$\mathbb{E}\left\{\sum_{k=0}^{\infty} \bar{I}_u(\mathbf{x}_k, \mu, \mathbf{w}_k)\right\} \leq \Delta_u. \tag{14}$$

Since the purpose of input rate chance constraints consists of reducing the rate of change in actuation and avoiding bang-bang type control, it is preferable to replace $\bar{I}_u(\cdot)$ with its smooth approximation $I_u(\cdot)$, defined in Equation (11). This is a conservative approximation since it satisfies $I_u(\mathbf{x}, \mu(\mathbf{x}), \mathbf{w}) > \bar{I}_u(\mathbf{x}, \mu(\mathbf{x}), \mathbf{w}) \ \forall \{\mathbf{x}, \mu(\cdot), \mathbf{w}\}$. Therefore, Problem 1 is conservatively reformulated as follows:

Problem 2. Conservative Chance Constrained Optimal Altitude Control Problem

$$\min_{\mu} \quad \mathbb{E}\left\{\sum_{k=0}^{\infty} g(\mathbf{x}_k, \mu, \mathbf{w}_k)\right\} \tag{15a}$$

$$\text{s.t.} \quad \mathbb{E}\left\{\sum_{k=0}^{\infty} I_u(\mathbf{x}_k, \mu, \mathbf{w}_k)\right\} \leq \Delta_u \tag{15b}$$

$$\text{Eqs. (9c)(9d)(9e).} \tag{15c}$$

Note that we replaced the original input rate chance constraint by a constraint involving an infinite sum of indicator functions. In the following section, we replace Problem 2 by its dual problem and leverage Bellman's Principle of Optimality [4] to compute the values of the infinite sums in Equations (15a) and (15b).

3.2 Dual Problem and Risks Definitions

The dual of Problem 2 is given as

Problem 3. Dual Optimal Altitude Control Problem

$$\max_{\lambda \geq 0} \quad \min_{\mu} \mathbb{E}\left\{\sum_{k=0}^{\infty} \left(g + \lambda \cdot I_u\right)(\mathbf{x}_k, \mu, \mathbf{w}_k)\right\} - \lambda \cdot \Delta_u \tag{16a}$$

$$\text{s.t.} \quad \mathbf{x}_k, \mathbf{w}_k \text{ satisfy (1) } \forall k \geq 0 \tag{16b}$$

$$\mu(\mathbf{x}_k) \in \mathcal{U}(\mathbf{x}_k) \ \forall k \geq 0 \tag{16c}$$

$$\mathbf{x}_0 \in \mathcal{X}_0, \tag{16d}$$

with λ the dual variable corresponding to the input rate constraint. Note that for a given λ, the inner minimization problem in Equation (16a) can be rewritten using Bellman's Principle of Optimality [4] as

$$J^{\lambda}(\mathbf{x}) = \min_{\mu^{\lambda}} \mathbb{E}\left\{\sum_{k=0}^{\infty} \left(g + \lambda \cdot I_u\right)(\mathbf{x}_k, \mu^{\lambda}, \mathbf{w}_k) \,\middle|\, \mathbf{x}_0 = \mathbf{x},\ \mathbf{x}_{k+1} = f(\mathbf{x}_k, \mu^{\lambda}(\mathbf{x}_k)) + \mathbf{w}_k\right\} \tag{17a}$$

$$= \min_{\mu^{\lambda}} \mathbb{E}\left\{\left(g + \lambda \cdot I_u\right)(\mathbf{x}, \mu^{\lambda}, \mathbf{w}) + J^{\lambda}(\mathbf{x}^{+})\right\}, \tag{17b}$$

where $\mathbf{x}^{+} := f(\mathbf{x}, \mu^{\lambda}(\mathbf{x})) + \mathbf{w}$. By discretizing the state and input spaces, this equation can be solved using the VI method [4], by setting arbitrary initial values for $J^{\lambda}(\mathbf{x})$ and iteratively solving Equation (17) for all $\mathbf{x}$. In Section 3.5, we prove that this method always converges to a unique solution given any initial guess for $J^{\lambda}(\mathbf{x})$. To make the structure of the cost explicit, we define the following indicator function:

$$I_{crit}(\mathbf{x}_k) := \begin{cases} 1 & \text{if } \mathbf{x}_k \in \mathcal{X}_{\text{crit}} \\ 0 & \text{otherwise} \end{cases}, \tag{18}$$

Using $I_{crit}(\cdot)$ and $I_u(\cdot)$, we define $r^{\mu}_{crit}(\cdot)$ and $r^{\mu}_{u}(\cdot)$ the risks to respectively enter the critical region $\mathcal{X}_{\text{crit}}$ and to violate the input rate constraints from any state $\mathbf{x}_0$ as

$$r^{\mu}_{crit}(\mathbf{x}_0) := \mathbb{E}\left\{\sum_{k=0}^{\infty} I_{crit}(\mathbf{x}_k) \,\middle|\, \mathbf{x}_0, \mu\right\}, \quad r^{\mu}_{u}(\mathbf{x}_0) := \mathbb{E}\left\{\sum_{k=0}^{\infty} I_u(\mathbf{x}_k) \,\middle|\, \mathbf{x}_0, \mu\right\}. \tag{19}$$

Given a policy μ and using the VI method, they can be computed independently by evaluating:

$$r^{\mu}_{crit}(\mathbf{x}_0) = \mathbb{E}\left\{I_{crit}(\mathbf{x}_0) + r^{\mu}_{crit}(f(\mathbf{x}_0, \mu(\mathbf{x}_0)) + \mathbf{w}_0)\right\} \tag{20a}$$

$$r^{\mu}_{u}(\mathbf{x}_0) = \mathbb{E}\left\{I_u(\mathbf{x}_0) + r^{\mu}_{u}(f(\mathbf{x}_0, \mu(\mathbf{x}_0)) + \mathbf{w}_0)\right\}. \tag{20b}$$

Using these functions and the definition of the original cost $g(\cdot)$ in Equation (8), we note that for a given control policy μ^{λ} and dual variable λ, the cost in Equation (17) can be rewritten as

$$J^{\lambda}(\mathbf{x}_0) = \mathbb{E}\left\{\alpha_h (z_N - h_{goal})^2 \,\middle|\, \mathbf{x}_0, \mu^{\lambda}\right\} + r^{\mu^{\lambda}}_{crit}(\mathbf{x}_0) + \lambda \cdot r^{\mu^{\lambda}}_{u}(\mathbf{x}_0), \tag{21}$$

with z_N the altitude at apogee. This reformulation explicitly shows that minimizing the cost in Equation (16a) for an appropriate λ minimizes the sum of the risk to enter $\mathcal{X}_{\text{crit}}$ and of the deviation to the target apogee while respecting input rate chance constraints.

To ensure the satisfaction of input rate chance constraints for the entire flight, starting from the state of the rocket on the launchpad $\mathbf{x}_{pad}$, it is possible to use the values of the risks $r_u^\mu(\mathbf{x}_0)$ from all burnout states $\mathbf{x}_0 \in \mathcal{X}_0$. Since the airbrakes are retracted until motor burnout at $\mathbf{x}_0 \in \mathcal{X}_0$, this enables the computation of a solution on a discretized state space which is smaller than if all states $\mathbf{x}_k \in \mathbb{R}^2$ had to be considered. To compute $r_u^\mu(\mathbf{x}_{pad})$ and $r_{crit}^\mu(\mathbf{x}_{pad})$, we use the risk values for all burnout states $\mathbf{x}_0 \in \mathcal{X}_0$ and weight the contribution of each term by its probability $\Pr(\mathbf{x}_0|\mathbf{x}_{pad})=p(\boldsymbol{\zeta}_j)=1/M_0$, with $\mathcal{X}_0$ and M_0 defined in Section 2.3. Indeed, using (a) Bellman's Principle of Optimality [4], (b) the law of the Unconscious Statistician [10] and $I_{crit}(\mathbf{x}_{pad}) = 0$, we obtain

$$r_{crit}^\mu(\mathbf{x}_{pad}) \overset{(a)}{=} \mathbb{E}\left\{I_{crit}(\mathbf{x}_{pad}) + r_{crit}^\mu(f_0(\mathbf{x}_{pad}, \boldsymbol{\zeta}_j))\right\} \overset{(b)}{=} \sum_{\boldsymbol{\zeta}_j} r_{crit}^\mu(f_0(\mathbf{x}_{pad}, \boldsymbol{\zeta}_j)) \cdot p(\boldsymbol{\zeta}_j)$$

$$= 1/M_0 \cdot \sum_{\mathbf{x}_0 \in \mathcal{X}_0} r_{crit}^\mu(\mathbf{x}_0), \tag{22a}$$

$$r_u^\mu(\mathbf{x}_{pad}) = 1/M_0 \cdot \sum_{\mathbf{x}_0 \in \mathcal{X}_0} r_u^\mu(\mathbf{x}_0). \tag{22b}$$

Note that assuming that a control input policy $\mu(\mathbf{x})$ is computed such that the input rate chance constraint is satisfied ($r_u^\mu(\mathbf{x}_{pad}) \leq \Delta_u$), the critical risk $r_{crit}^\mu(\mathbf{x}_{pad})$ provides a measure of the probability of success of the flight. In the following section, we use these risk definitions and present an algorithm to solve Problem 3, ensuring the satisfaction of input rate chance constraints for the entire flight from $\mathbf{x}_{pad}$.

3.3 Algorithm

We present an algorithm to solve Problem 3 offline, such that the solution $\mu(\mathbf{x})$ can be stored in memory and used in real time as a look-up table. Compared to DP formulations as in [20] and since the problem was defined as an infinite horizon optimal control problem, the resulting control input policy $\mu(\mathbf{x})$ is time-invariant and does not require an estimate of the final time at apogee.

As presented in the previous section, for a fixed dual variable λ, it is possible to compute the optimal cost $J^\lambda(\cdot)$, control policy $\mu^\lambda(\cdot)$ and resulting risk $r_u^{\mu^\lambda}(\mathbf{x}_{pad})$ using the VI method. To find the optimal value λ^* ensuring the satisfaction of the input rate chance constraint, we use a root-finding algorithm as in [20]. Intuitively, our algorithm consists of sequentially computing $J^\lambda(\cdot)$ and $\mu^\lambda(\cdot)$ for a fixed λ, computing the resulting risk $r_u^{\mu^\lambda}(\cdot)$ and updating λ using the bisection method to satisfy the input rate chance constraint $r_u^{\mu^\lambda}(\mathbf{x}_{pad}) < \Delta_u$. Since $J^\lambda(\cdot)$ is concave in λ and r_u^μ is monotonically not increasing with respect to λ, any root-finding algorithm is guaranteed to converge to a solution approximating the optimum of the original primal problem 1, up to the duality gap and the approximation error of the solution of the dual problem 3 [20].

First, lines 1−5 check if the problem is feasible without the input rate constraint. If the chance constraint is satisfied, then the solution is returned and the algorithm terminates. Otherwise, the algorithm successively updates λ in step 9 and solves the dual problem in step 10 using the VI Method. The input risk is then computed in step 11 and if $r_u^{\mu^\lambda} = \Delta_u$, the algorithm terminates and returns the optimal solution μ^λ. Otherwise, we stop after i_{max} iterations and return λ_{high}, guaranteeing that the error to the optimal dual variable λ^* for Problem 3 is bounded as $|\lambda_{high} - \lambda^*| < \lambda^+/2^{i_{max}}$. Note that another root-finding algorithm and stopping criterion as in [20] can also be used. The initial maximum value for λ is initialized with a value λ^+ chosen to satisfy the input rate risk constraint $r_u^{\mu^0}(\mathbf{x}_{pad}) < \Delta_u$.

3.4 Discretization of the State-Input Spaces

In order to apply to the Value Iteration method to solve Equations (17b) and (20b), it is necessary to discretize the state and input spaces $\mathbb{R}^2 \times \mathbb{R}$. Although general techniques exist to optimize the discretization schemes of continuous spaces [24], we opt for an approach leveraging the structure of the problem. From the definition of the statespace and to minimize the deviation to the apogee altitude, we constrain the optimal action for any state inside the critical region $\mathcal{X}_{crit}$ as $\mu(\mathbf{x}) = 1$ if $z \geq \gamma_{max}(v)$ and $\mu(\mathbf{x}) = 0$ if $z \leq \gamma_{min}(v)$. Therefore, it suffices to only include states in the controllable region $\mathcal{X}$ in the discretized grid, and solve

Algorithm 1 Risk Minimization with Input Rate Chance Constraints

Input. Maximum input rate b_u, probability threshold Δ_u, maximum number of iterations i_{max}.
Output. Feasible optimal policy μ^*

Check if problem is feasible with $\lambda = 0$ (no input cost)
  Compute $\mu^0 = \text{argmin}_\mu \mathbb{E}\left\{\sum_k \left(I_{crit} + g_{apg}\right)(\mathbf{x}_k)\right\}$
  Compute input rate risk $r_u^{\mu^0}(\mathbf{x}_0)$
if $r_u^{\mu^0}(\mathbf{x}_{pad}) < \Delta_u$ **then**
  return μ^0
$\{i, \lambda_{low}, \lambda_{high}\} \leftarrow \{0, 0, \lambda^+\}$
while $i < i_{max}$ **do**
  $i \leftarrow i + 1$
  $\lambda \leftarrow \text{bisection}(\lambda_{low}, \lambda_{high})$
  $\left\{J^\lambda, \mu^\lambda\right\} \leftarrow$ ValueIteration(Equation (17))
  Compute input rate risk $r_u^{\mu^\lambda}(\mathbf{x})$
  if $r_u^{\mu^\lambda}(\mathbf{x}_{pad}) = \Delta_u$ **then**
    return μ^λ
  else if $r_u^{\mu^\lambda}(\mathbf{x}_{pad}) < \Delta_u$ **then**
    $\lambda_{high} \leftarrow \lambda$
  else if $r_u^{\mu^\lambda}(\mathbf{x}_{pad}) > \Delta_u$ **then**
    $\lambda_{low} \leftarrow \lambda$
return $\mu^{\lambda_{high}}$

the Bellman Equations (17b) and (20b) subject to the boundary conditions above and in Equation (7). To optimize the resolution of this grid, it is split linearly into 200 different velocity values from zero to the maximum controllable velocity v_{max}. Also, it is defined with a different scale of the height h for each velocity v, as shown in Figure 3. For each v, the height h is discretized into 200 uniformly distributed points from the critical minimum height $\gamma_{min}(v)$ to the critical maximum height $\gamma_{max}(v)$. As a state $\mathbf{x}$ could lie between grid points, bilinear interpolation is used to interpolate its optimal control and cost values from the neighbouring grid cells. To improve the accuracy of the interpolation, each curved horizontal segment between two vertical grid lines is further subdivided into 10 linear segments, as shown in the close view of Figure 3. Note that this additional subdivision is only used for interpolation and no optimal control and cost values are computed at those nodes. The control input space $\mathcal{U}$ is discretized into 256 values uniformly distributed in $[0, 1]$, which is the maximum number of values that each control value u can represent as a byte.

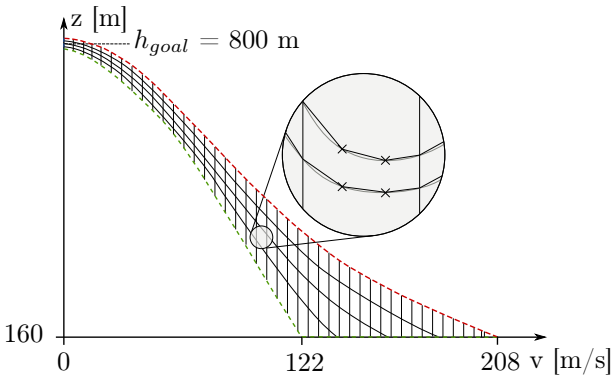


Figure 3: Discretization of $\mathcal{X}$ using a constant scale for velocities and a velocity-dependent scale for the heights. The close view shows the additional subdivision of velocities to improve the accuracy of the bilinear interpolation method.

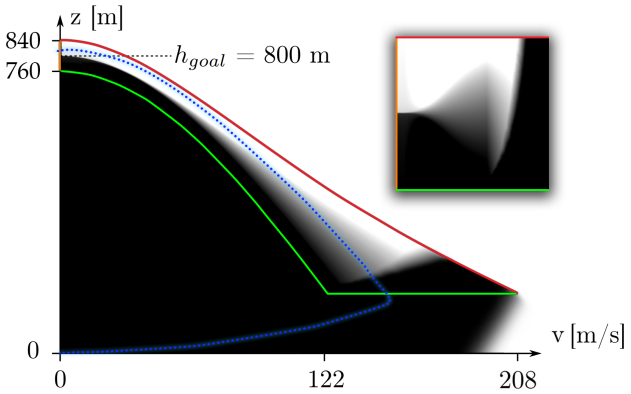


Figure 4: Control table computed by the algorithm, with a gray scale denoting the optimal airbrakes extension, with u=1 in white and u=0 in black. In blue is the trajectory of the rocket during the test launch. The solution is saved in memory as a squared look-up table (top right).

3.5 Proof of Convergence and Optimality

In this section, we show that the infinite horizon cost of Problem 3 is finite and the VI Method is assured to find a unique solution for Equations (17, 20a, 20b). First, due to definitions of the statespace in Section 2.3, any state $\mathbf{x}_k \in \mathcal{X}$ follows the dynamics in Equation (1) and transitions with a non-zero probability to either $\mathcal{X}$, $\mathcal{X}_{crit}$ or to $\mathcal{X}_{apg}$, whereas any state $\mathbf{x}_N \in (\mathcal{X}_{apg} \cup \mathcal{X}_{\text{crit}})$ transitions with probability 1 to $\mathcal{X}_{end}$, where it remains $\forall k \geq N+1$. Therefore, Problem 3 is equivalent to a Markov model which can be represented as in Figure 5:

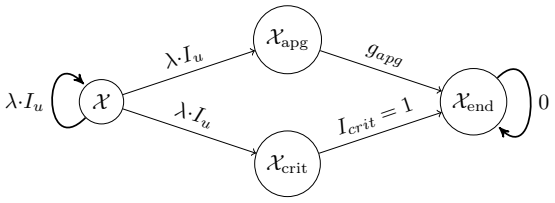


Figure 5: Markov chain model for Problem 3. Arrows denote possible transitions with their corresponding cost above.

Due to the end region $\mathcal{X}_{end}$ from which no additional cost occurs and to the discretization of the state-input spaces (Section 3.4), this problem is equivalent to a stochastic shortest path problem with a finite number of states and inputs. Also, any policy is proper due to the dynamics[3]. Furthermore, the problem is feasible since there exists policies (such as $\mu(\mathbf{x})$=0 $\forall \mathbf{x}$) which satisfy the input constraint. Hence, using the Existence and Unicity of Solutions Theorem [4], the solution of the Bellman Equation in Equations (17,20a,20b) is unique and the VI Method converges to their optimal value.

4. Results

4.1 Implementation

The VI method was implemented in C++. The high fidelity simulator for noise identification was written in Matlab. The flight software was written in C.

The solution of Bellman Equations (17) and (20b) are efficiently computed in two steps. First, the expectation operator $\mathbb{E}$ is performed by convolving Gaussian kernels representing the probability distributions $\mathcal{N}(\boldsymbol{\mu}_w(\mathbf{x}), \boldsymbol{\Sigma}_w(\mathbf{x}))$ of each state $\mathbf{x}$, which are scaled to match the discretization grid shown in Figure 3. This operation is approximated using three box kernel and an offset for each point as in [17], reducing the computational complexity from $\mathcal{O}(nr^2)$ to $\mathcal{O}(n)$ with n the statespace dimension and r the radius of the kernel. Secondly, the optimal control input policy $\mu(\mathbf{x})$ for each state $\mathbf{x}$ is computed by evaluating the cost for each feasible control input $u \in \mathcal{U}$. To efficiently iterate over all possible control inputs, the transition vectors $f(\mathbf{x}, u)$ are precomputed $\forall(\mathbf{x}, u) \in \mathcal{X} \times \mathcal{U}$ by integrating the dynamics in Equation (23) using a fifth order Adams-Bashforth integration method with Δt=0.4 s. The final solution is saved in memory as a lookup table, requiring 40KB only. For a given state $\mathbf{x}$ estimated during the flight, the optimal control input is bilinearly interpolated from the values of the states in the grid which are adjacent to $\mathbf{x}$, as described in Section 3.4.

4.2 Hardware

A model rocket was designed and built to demonstrate the proposed controller. The rocket is 1.85m long, has a diameter of 0.1m and weights 4.25kg without the propellant. The motor is an AeroTech J460 with an average thrust of 460 N and a total impulse of 805.5 N s developed during a burn of 1.8 s generated by 415 g of fast burning Blue Thunder propellant. The main body of the rocket is split into two parts connected by the airbrakes section, as shown in Figure 1. The three airbrake plates have an exposed surface area of 11.5cm^2 each at their maximum extension of 2.1cm. They are mounted on linear guides and are all controlled by a single servomotor Futaba S3072HV with absolute positioning capabilities, for which the full airbrake deployment time was measured to be 0.25 seconds. The servomotor is connected to a Raspberry Pi Model A+

[3] I.e. any policy leads the state to $\mathcal{X}_{end}$ since the rocket slows down due to gravity and drag and any state $\mathbf{x}$ with $v < 0$ transitions to $\mathcal{X}_{end}$.

running a state estimation algorithm fusing the accelerometer and gyroscope measurements from a MPU6050 and the altimeter readings from a BMP180. Two Altimax Simply altimeters are used for recovery and to verify the final apogee altitude.

4.3 Test Launch

The altitude optimal control problem is defined with a target altitude $h_{goal} = 800$m. During a test flight on October 6, our rocket reached approximately 808.2m as shown on Table 4. The final apogee estimates from the three pressure sensors are reported in the table. Since the rocket is mostly stationary at apogee, there is no measurement bias in the altimeter measurements which makes their measurement more reliable than the fusion of the IMU with the altimeter sensor.

	Apogee Altitude	Relative Error
BMP180	809.5m ± 0.7m	1.2%
AltiMAX 1	807m ± 0.5m	0.9%
AltiMAX 2	808m ± 0.5m	1.0%
Mean	808.2m ± 0.6m	1.0%

Table 4: Measured apogee altitude for different sensors and relative errors for a target of 800m.

Figure 6 shows the control input u during the flight and the measured change in the drag coefficient, computed as

$$C_d(t) = \frac{2m(\dot{v}(t) + g)}{\rho(z(t))v(t)^2 A_{ref}},$$

where m is the rocket mass after motor burnout, $\rho(\cdot)$ is the atmospheric pressure, g the gravitational acceleration, A_{ref} the reference area and z, v and $\dot{v}$ are those estimated and recorded during the flight. The simulated C_d values in Figure 6 are computed as a function of u, v using Equation (28) and the properties of the rocket, described in the appendix. Also, the input rate constraint is well respected, despite the peak at $t = 2.2s$.

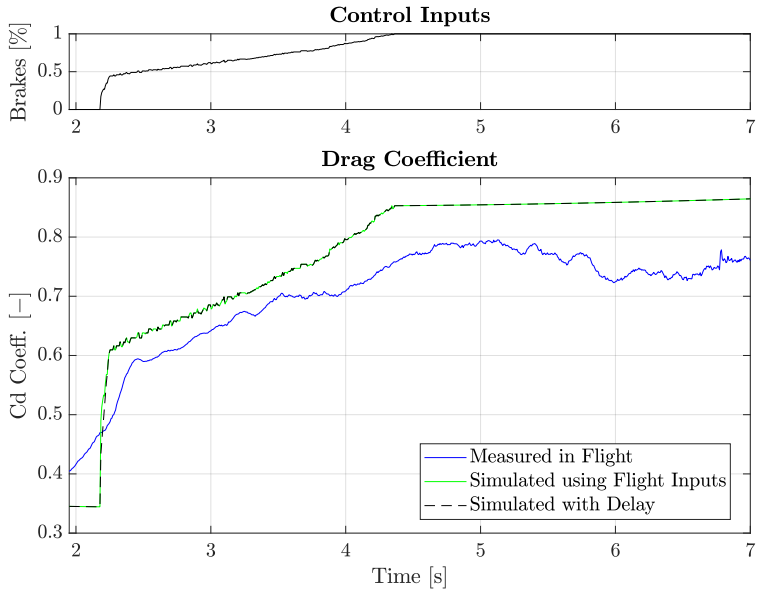


Figure 6: Simulated and measured drag coefficient. Despite model missmatch between simulated and measured drag, the airbrakes caused the rocket to reach the target apogee and the input rate constraints were satisfied at all times.

5. Conclusion

This work addressed the problem of altitude control of a rocket. We presented an algorithm solving a chance-constrained infinite horizon problem and a method to identify a model of disturbances capturing the effects of model missmatch, parameter uncertainty and external disturbances. To include input rate chance constraints, a new approach was proposed which is suitable to any problem formulation using Bellman Equation, such as DP and Reinforcement Learning. This problem was solved offline using an algorithm for which convergence and optimality guarantees were provided. The solution was saved in memory as a look-up table from which control inputs were accessed and used in real time. Our algorithm was successfully verified in a test flight, with an apogee error of less than 1% for a target apogee of 800m. The problem formulation and resulting algorithm proposed in this work could be used for multiple aerospace applications, such for the entry, descent and landing problem and the design of flight trajectories.

Acknowledgments

The authors thank ARIS, its sponsors and members for their support in the construction and funding of the rocket and of this manuscript, Prof. Dr. Maryam Kamgarpour for her insightful advice during this project, Hassan Arif for sharing his work on the simulation of the rocket and Joseph Lorenzetti for his valuable feedback.

A. Rocket Dynamics

This appendix presents the 6 DoF dynamical model of the rocket. This model is used both for the verification of the controller, and to identify disturbances and burnout positions using Monte Carlo simulations.

A.1 Variables and Parameters Definition

Variables

z	nominal height
v	nom. vertical velocity and accel.
$\dot{v}$	vertical acceleration
$\mathbf{x}$	rocket state
$\mathbf{r}$	position of CoM
$\mathbf{q}$	attitude quaternion
$\mathbf{v}, \boldsymbol{\omega}$	linear and angular velocities
$\mathbf{F}_g$	gravitational force
$\mathbf{F}_A$	axial aerodynamic force
$\mathbf{F}_N$	normal aerodynamic force
$\boldsymbol{\tau}_N$	normal torque
$\boldsymbol{\tau}_{damp}$	thrust damping torque
α	angle of attack
Ma	Mach number
$\mathbf{e}_{roll}$	longitudinal/roll axis
$\mathbf{r}_{CoP}$	location of center of pressure
$\mathbf{r}_{CoM}$	location of center of mass
C_A	axial drag coefficient
C_N	normal drag coefficient
$\mathbf{v}_{app}$	apparent velocity
$\mathbf{v}_{wind}$	wind velocity
α	angle of attack
$\mathbf{R}$	rotation matrix

Environment Variables and Parameters

$\mathbf{g}$	std. gravitational acceleration
$\rho(z)$	air density
$\sigma_{wind}(z)$	wind speed std. dev.
w_*	convective velocity scale
z_i	average mixed layer depth

Rocket Parameters

m	mass at burnout	$4.2525\,\mathrm{kg}$
$\mathbf{I}$	diagonal inertia at burnout ($I_x, I_y = I_z$)	$0.035, 4.6\,\mathrm{kg\,m^2}$
L_{rocket}	rocket length	$1.842\,\mathrm{m}$
θ_{pad}	launch rail angle	$90°$
A_{ref}	reference area	$81.7\times10^{-4}\,\mathrm{m^2}$
A_{brakes}	control surface area	$34.5\times10^{-4}\,\mathrm{m^2}$
CD_0	nom. brakes drag coeff.	1.17
r_{brakes}	airbrakes position	$1.54\,\mathrm{m}$
h_{brakes}	max. airbrakes length	$0.021\,\mathrm{m}$

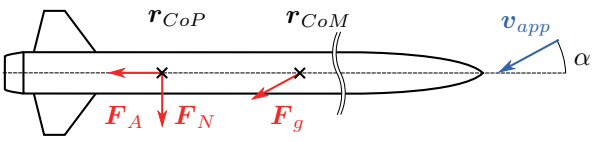
Parameters at burnout, $u = 0, \alpha = 0$

z	height	$232\,\mathrm{m}$
$\lvert\mathbf{v}\rvert$	velocity	$166\,\mathrm{m\,s^{-1}}$
$\bar{x}_{stab}$	stability margin	$0.408\,\mathrm{m}$
C_A	axial drag coefficient	0.35
C_{N_α}	normal drag coeff. derivative	13.6

Parameters at burnout, $u = 1, \alpha = 0$

C_A	axial drag coefficient	0.79
C_{damp}	damping moment drag coeff.	4.88

Forces Diagram



A.2 Dynamics

The rocket is modeled as a rigid body of mass m and inertia matrix $\mathbf{I}$. It is described by its state $\mathbf{x} = [\mathbf{r}; \mathbf{q}; \mathbf{v}; \boldsymbol{\omega}] \in \mathbb{R}^{12}$, with $\mathbf{r} = [x; y; z] \in \mathbb{R}^3$ the position of the center of mass, $\mathbf{q} = [q_w, \mathbf{q}_v] \in \mathbb{R}^4$ the quaternion describing its orientation, $\mathbf{v} \in \mathbb{R}^3$ the linear velocity and $\boldsymbol{\omega} \in \mathbb{R}^3$ the angular velocity. The system is actuated using the control inputs $u \in \mathcal{U} = [0, 1]$, representing the airbrakes extension. To simplify notations, we dropped the dependency on the time t and all quantities are expressed in an inertial frame which origin is aligned with the launchpad with the z-axis pointing upwards. Also, as control of the rocket is allowed only after motor burnout, no motor thrust is acting and the mass and inertia are constant. The evolution of the system is thus governed by its dynamics $f(\cdot)$ with uncertain parameters $\boldsymbol{\zeta} \in \mathbb{R}^p$ as

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{r}} \\ \dot{\mathbf{q}} \\ \dot{\mathbf{v}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} = f(\mathbf{x}, \mathbf{u}, \boldsymbol{\zeta}) = \begin{bmatrix} \mathbf{v} \\ [\frac{1}{2}(\boldsymbol{\omega} \cdot \mathbf{q}_v);\ \frac{1}{2}(q_w\boldsymbol{\omega} + (\boldsymbol{\omega} \times \mathbf{q}_v))] \\ \frac{1}{m}\left(\mathbf{F_g} + \mathbf{F}_A(u) + \mathbf{F}_N\right) \\ \mathbf{I}^{-1}\left(\boldsymbol{\tau}_N + \boldsymbol{\tau}_{damp}\right) \end{bmatrix}, \tag{23}$$

with $\mathbf{F_g}, \mathbf{F}_A(u), \mathbf{F}_N$ the gravitational, axial drag and normal drag forces and $\boldsymbol{\tau}_N, \boldsymbol{\tau}_{damp}$ the normal and damping drag torques. To compute all forces and torques, we assume subsonic speeds and small angles of attack α. Also, the rocket is assumed to be symmetrical along its longitudinal roll axis $\mathbf{e}_{roll}$ with respect to its mass distribution and aerodynamical forces. $\mathbf{e}_{roll}$ thus passes through the centers of mass $\mathbf{r}_{CoM} := \mathbf{r}$ and of pressure $\mathbf{r}_{CoP}$, defined in Equation (32). Denoting the Euclidean norm of a vector $\mathbf{v} \in \mathbb{R}^n$ as $\|\mathbf{v}\|$, the forces and torques are defined as

$$\mathbf{F_g} = m\mathbf{g} \qquad \text{Gravity} \tag{24a}$$

$$\mathbf{F}_A(u) = -\frac{\rho(z)}{2} A_{ref} C_A(\mathbf{x}, u) \|\mathbf{v}_{app}\|^2 \cdot \mathbf{e}_{roll} \qquad \text{Axial} \tag{24b}$$

$$\mathbf{F}_N = \frac{\rho(z)}{2} A_{ref} C_N(\mathbf{x}) \|\mathbf{v}_{app}\|^2 \cdot \left(\mathbf{e}_{roll} \times (\mathbf{e}_{roll} \times \frac{\mathbf{v}_{app}}{\|\mathbf{v}_{app}\|})\right) \qquad \text{Normal} \tag{24c}$$

$$\boldsymbol{\tau}_N = \bar{\mathrm{x}}_{stab} \|\mathbf{F}_N\| \cdot (\mathbf{e}_{roll} \times \mathbf{v}_{app}) \qquad \text{Normal} \tag{24d}$$

$$\boldsymbol{\tau}_{damp} = -C_{damp}(\mathbf{x})(\mathbf{R}\,\mathrm{diag}[1, 1, 0]\,\mathbf{R}^{-1})\boldsymbol{\omega}, \qquad \text{Damping} \tag{24e}$$

with $\mathbf{g}$ the standard gravitational acceleration, $\rho(z)$ the air density, A_{ref} the reference area, α the angle of attack, $\mathbf{v}_{app}$ the apparent velocity, $\mathbf{e}_{roll}$ the longitudinal roll axis of the rocket, $\bar{\mathrm{x}}_{stab} = \|\mathbf{r}_{CoP} - \mathbf{r}_{CoM}\|$ the stability margin, $\mathbf{R}$ the rotation matrix transforming a vector in body frame to the inertial frame, $\mathrm{diag}[1, 1, 0]$ a diagonal matrix with $\{1, 1, 0\}$ as diagonal elements, and C_A, C_N, C_{damp} the axial, normal and damping drag coefficients, defined in section A.3. $\boldsymbol{\tau}_{damp}$ in Equation (24e) is computed as in [7]. Velocity-dependent terms are defined as

$$\mathbf{v}_{app} = \mathbf{v}_{CoP} + \mathbf{v}_{wind}(z) \qquad \text{Apparent Velocity of CoP} \tag{25a}$$

$$\mathbf{v}_{CoP} = \mathbf{v}_{CoM} + \boldsymbol{\omega} \times (\mathbf{r}_{CoP} - \mathbf{r}_{CoM}) \qquad \text{Velocity of CoP} \tag{25b}$$

$$\alpha = \cos^{-1}\left(\frac{\mathbf{v}_{app}}{\|\mathbf{v}_{app}\|} \cdot \mathbf{e}_{roll}\right), \qquad \text{Angle of Attack} \tag{25c}$$

with $\mathbf{v}_{CoP}$ and $\mathbf{v}_{wind}(z)$ the center of pressure and wind velocities, α the angle of attack. $\rho(z)$ is computed according to the US Standard Atmosphere 1976 [19]. We model the wind velocity as following a zero-mean Gaussian distribution as

$$\mathbf{v}_{wind}(z) := [0;\ 0;\ v_{wind,z}(z)], \quad v_{wind,z}(z) \sim \mathcal{N}\left(0, \sigma^2_{wind}(z)\right), \tag{26}$$

with the following empirical equation from [23] for the variance

$$\sigma^2_{wind}(z) = 1.8 \cdot w_*^2 \cdot \left(\frac{z}{z_i}\right)^{2/3} \cdot \left(1 - 0.8\frac{z}{z_i}\right)^2, \tag{27}$$

with w_* the convective velocity scale, set to 2.0 m/s [25] and z_i the average mixed layer depth, set to 500m [14]. This corresponds to a zero-mean vertical turbulent flow in the Atmospheric Boundary Layer. This is to contrast with work such as [11] which is only interested in horizontal wind. Note that this model of the wind gives low variances for high altitudes and that the variance is zero at ground level.

A.3 Drag Coefficients

In this section, the drag aerodynamic coefficients used to compute aerodynamic forces and torques in Equations (24) are defined. The values of these coefficients were verified experimentally in wind tunnel tests.

First, the total axial drag coefficient $C_A(\mathbf{x}, u)$ is computed using a weighted average of the axial drag coefficients of the rocket $C_{A,rocket}$ and of its airbrakes $C_{A,brakes}$, normalized by their respective areas A_{ref}, A_{brakes} as

$$C_A(\mathbf{x}, u) = C_{A,rocket} + u\,\frac{A_{brakes}}{A_{ref}}\,C_{A,brakes}, \quad u \in [0, 1]. \tag{28}$$

The axial drag coefficient of the rocket $C_{A,rocket}(\mathbf{x})$ is computed according to the DATCOM method, originally introduced by Mandell [18]. It includes the contributions of the drag from the body $C_{D,fb}$, the base $C_{D,b}$, the fins $C_{D,f}$ and the interference between fins and body C_{D_i} of the rocket. Based on the assumption of a small angle of attack, the drag force is assumed to be approximately aligned with the longitudinal axis of the rocket $\mathbf{e}_{roll}$. After applying an additional correction for the compressibility of the air flow as in [7], with Ma the Mach number, the axial drag coefficient is computed as

$$C_{A,rocket}(\mathbf{x}) \approx (C_{D,fb} + C_{D,b} + C_{D,f} + C_{D_i} + C_{D,add})\,\frac{1}{\sqrt{1 - Ma^2}}. \tag{29}$$

To include the drag due to the deployment of the airbrakes, the control surfaces are approximated as rectangular plates with a maximum extension length h_{brakes} and area A_{brakes} at a position r_{brakes} along the rocket. The area-normalized axial drag coefficient of the airbrakes $C_{A,brakes}$ can thus be computed according to [16, 3-17] as

$$C_{A,brakes} = C_{D0}\left(1 - 0.25\frac{\delta}{h_{brakes}}\right), \tag{30}$$

where C_{D0} is the nominal drag coefficient of a quadratic plate and δ denotes the boundary layer thickness of the air flow around the airbrakes, which depends on the local Reynolds number as defined in [18].

Second, according to Barrowman [2], the normal force coefficient C_N can be computed as the product of the angle of attack α with the sum of the body $C_{N_\alpha,B}$, nosecone $C_{N_\alpha,N}$, fins $C_{N_\alpha,F}$ and fins-body interference $C_{N_\alpha,T(B)}$ drag coefficient derivatives, with a correction term $C_{N_{\alpha^2}}$ defined by Galejs [13] to account for body lift, as

$$C_N(\mathbf{x}) = C_{N_\alpha}\,\alpha \tag{31a}$$

$$C_{N_\alpha} = C_{N_\alpha,B} + C_{N_\alpha,N} + C_{N_\alpha,F} + C_{N_\alpha,T(B)} + C_{N_{\alpha^2}}. \tag{31b}$$

As the airbrakes are symmetrical with respect to the longitudinal axis of the rocket and are actuated with a single motor, no normal drag is induced by the airbrakes.

Finally, the damping moment drag coefficient $C_{damp}(\mathbf{x})$ is determined according to [18, 202]. To compute the normal torque $\boldsymbol{\tau}_N$ using the stability margin $\bar{\mathrm{x}}_{stab}$, it is necessary to compute the position of the center of pressure $\mathbf{r}_{CoP}$. Given the respective centers of pressure $\mathbf{r}_{CoP,i}$ of the different sections of the rocket (body, nosecone, ...) [2], it is computed as a weighted sum of each drag coefficient derivative $C_{N_\alpha,i}$ as

$$\mathbf{r}_{CoP} = \frac{\sum \mathbf{r}_{CoP,i}\,C_{N_\alpha,i}}{\sum C_{N_\alpha,i}}. \tag{32}$$

A.4 Monte Carlo simulations

The random parameters $\boldsymbol{\zeta} = [f_{motor}, f_{C_D}, \theta_{pad}, \mathbf{v}_{wind}]$ incorporate the effects of different motor efficiencies, aerodynamic coefficient correction factors, launchpad rail headings and wind velocities where the velocity at each altitude $v_{wind}(z)$ follows Equation (26). The other parameters are drawn according to $f_{motor} \sim \mathrm{Unif}(90\%, 110\%)$, $f_{C_D} \sim \mathrm{Unif}(95\%, 105\%)$ and $\theta_{pad} \sim \mathrm{Unif}(89°, 91°)$, where $\mathrm{Unif}(a, b)$ denotes the uniform distribution with values in $[a, b]$. To account for possible drag coefficient modeling errors, we simulate the system by replacing $C_A(\mathbf{x}, u)$ with $f_{C_D}C_A(\mathbf{x}, u)$ to compute axial drag forces in Equation (24b). To compute burnout positions and simulate the system, the rocket is initially on the launchpad and pointing upwards with a pitch angle θ_{pad}. Until motor burnout, it is subject to a thrust force $f_{motor}T(t)\,\mathbf{e}_{roll}$ and the varying thrust, mass and inertia $T(t), m(t), \mathbf{I}(t)$ depend on the thrust curve of the motor[4].

[4] For the AeroTech J460 motor mounted on the rocket, the thrust curve used in the simulator can be found on `thrustcurve.org`.

References

[1] H. Arellano-Garcia and G. Wozny. Chance constrained optimization of process systems under uncertainty: I. Strict monotonicity. *Computers & Chemical Engineering*, 33(10), 2009.

[2] J. S. Barrowman and J. A. Barrowman. The Theoretical Prediction of the Center of Pressure. 1966.

[3] A. Bemporad and M. Morari. Robust model predictive control: A survey. In A. Garulli and A. Tesi, editors, *Robustness in identification and control*, pages 207–226. Springer London, 1999.

[4] D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 2nd edition, 2000.

[5] D. Bienstock, M. Chertkov, and S. Harnett. Chance-constrained optimal power flow: Risk-aware network control under uncertainty. *SIAM Review*, pages 461–495, 2014.

[6] L. Blackmore, M. Ono, and B. C. Williams. Chance-Constrained Optimal Path Planning With Obstacles. *IEEE Transactions on Robotics*, 27(6), 2011.

[7] Simon Box, Christopher M. Bishop, and Hugh Hunt. A stochastic six-degree-of-freedom flight simulator for passively controlled high power rockets. *Journal of Aerospace Engineering*, 24, 2010.

[8] J.-B. Caillau, M. Cerf, A. Sassi, E. Trélat, and H. Zidani. Solving chance constrained optimal control problems in aerospace via Kernel Density Estimation. *Optimal Control Applications and Methods*, 39(5):1833–1858, 2018.

[9] R. Chai, A. Savvaris, A. Tsourdos, S. Chai, and Y. Xia. A review of optimization techniques in spacecraft flight trajectory design. *Progress in Aerospace Sciences*, 2019.

[10] M.H. DeGroot and M.J. Schervish. *Probability and Statistics*. Addison-Wesley, 2012.

[11] Brandon Douglas Luders, Aaron Cole Ellertson, Jonathan How, and Ian Sugel. Wind uncertainty modeling and robust trajectory planning for autonomous parafoils. *Journal of Guidance, Control, and Dynamics*, 39:1–17, 2016.

[12] M. Farina, L. Giulioni, and R. Scattolini. Stochastic linear Model Predictive Control with chance constraints - A review. *Journal of Process Control*, 44:53–67, 2016.

[13] R. Galejs. Wind Instability - What Barrowman Left Out, 2018.

[14] J. R. Garratt. Review: the atmospheric boundary layer. *Earth Science Reviews*, 37:89–134, 1994.

[15] F. Gavilan, R. Vazquez, and E. F. Camacho. Chance-constrained model predictive control for spacecraft rendezvous with disturbance estimation. *Control Engineering Practice*, 20(2):111–122, 2012.

[16] S.F. Hoerner. *Fluid-Dynamic Drag: Theoretical, Experimental and Statistical Information*. Hoerner Fluid Dynamics, 1965.

[17] W. Jarosz. Fast image convolutions. *Association for Computing Machinery, Special Interest Group for Graphics, University of Illinois at Urbana-Champaign*, 2001.

[18] G.K. Mandell, G.J. Caporaso, and W.P. Bengen. *Topics in Advanced Model Rocketry*. MIT Press, 1973.

[19] U.S. Committee on Extension of the Standard Atmosphere. *U.S. Standard Atmosphere*. U.S. Government Printing Office, 1976.

[20] M. Ono, M. Pavone, Y. Kuwata, and J. Balaram. Chance-constrained dynamic programming with application to risk-aware robotic space exploration. *Autonomous Robots*, 39(4), 2015.

[21] H. Park, R. Zappulla, Z. Costantinos, J. Virgili-Llop, and M. Romano. Nonlinear model predictive control for spacecraft rendezvous and docking with a rotating target. In *Proceedings of the 27th AAS/AIAA Spaceflight Mechanics Meeting*, 2017.

[22] M. Sagliano, E. Mooij, and S. Theil. Onboard Trajectory Generation for Entry Vehicles via Adaptive Multivariate Pseudospectral Interpolation. *Journal of Guidance, Control, and Dynamics*, 40(2):466–476, 2017.

[23] R.B. Stull. *An Introduction to Boundary Layer Meteorology*. Atmospheric and Oceanographic Sciences Library. Springer Netherlands, 1988.

[24] W. T. B. Uther and M. M. Veloso. Tree based discretization for continuous state space reinforcement learning. In *Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence*. American Association for Artificial Intelligence, 1998.

[25] A. Venkatram. Estimating the convective velocity scale for diffusion applications. *Boundary-Layer Meteorology*, 15(4):447–452, 1978.

[26] Zinan Zhao and Mrinal Kumar. Split-Bernstein Approach to Chance-Constrained Optimal Control. *Journal of Guidance, Control, and Dynamics*, 40(11):2782–2795, 2017.