Computationally efficient stochastic MPC: a probabilistic scaling approach

Martina Mammarella[1], Teodoro Alamo[2], Fabrizio Dabbene[1], and Matthias Lorenzen[3]

Abstract— In recent years, the increasing interest in Stochastic model predictive control (SMPC) schemes has highlighted the limitation arising from their inherent computational demand, which has restricted their applicability to slow-dynamics and high-performing systems. To reduce the computational burden, in this paper we extend the probabilistic scaling approach to obtain low-complexity inner approximation of chance-constrained sets. This approach provides probabilistic guarantees at a lower computational cost than other schemes for which the sample complexity depends on the design space dimension. To design candidate simple approximating sets, which approximate the shape of the probabilistic set, we introduce two possibilities: i) fixed-complexity polytopes, and ii) ℓ_p-norm based sets. Once the candidate approximating set is obtained, it is scaled around its center so to enforce the expected probabilistic guarantees. The resulting scaled set is then exploited to enforce constraints in the classical SMPC framework. The computational gain obtained with the proposed approach with respect to the scenario one is demonstrated via simulations, where the objective is the control of a fixed-wing UAV performing a monitoring mission over a sloped vineyard.

I. INTRODUCTION

In recent years, the performance degradation of model predictive control (MPC) schemes in the presence of uncertainty has driven the interest towards stochastic MPC, to overcome the inherent conservativeness of robust approaches. A probabilistic description of the disturbance or uncertainty allows to optimize the average performance or appropriate risk measures. Furthermore, allowing a (small) probability of constraint violation, by introducing so-called chance constraints, seems more appropriate in some applications. As highlighted in [1], current SMPC methods can be divided in two main groups, depending on the approach followed to solve the chance-constrained optimization problem: (i) analytic approximation methods; and (ii) randomized [2] and scenario-based methods. For the analytic approximation methods, the probabilistic properties of the uncertainty are exploited to reformulate the chance constraints in a deterministic form. For the second class of methods, the craved

[1] Institute of Electronics, Computer and Telecommunication Engineering, National Research Council of Italy, Turin, Italy, martina.mammarella@ieiit.cnr.it, fabrizio.dabbene@ieiit.cnr.it
[2] Departamento de Ingeniería de Sistemas y Automtica, Universidad de Sevilla, Escuela Superior de Ingenieros, Camino de los Descubrimientos s/n, 41092 Sevilla, Spain, talamo@us.es
lorenzen@ist.uni-stuttgart.de

control performance and constraint satisfaction are guaranteed properly generating a sufficient number of uncertainty realizations and on the solution of a suitable constrained optimization problem, as proposed in [3], [4]. The main advantage of this class of stochastic MPC algorithms is given by the inherent flexibility to be applied to (almost) every class of systems, including any type of uncertainty and both state and input constraints, as long as the optimization problem is convex. On the other hand, they share two main drawback: i) slowness, which has limited their application to problems involving slow dynamics and where the sample time is measured in tens of seconds or minutes; and ii) a significant computational burden required for real-time implementation, narrowing the application domains to those involving low-computation assets. Some examples are [5] for water networks, [6] for river flood control, [7] for chemical processes, and [8] for energy plants.

An efficient solution to the aforementioned disadvantages was proposed in [9] where the SMPC controllers design is based on an *offline* sampling approach and only a predefined number of necessary samples are kept for online implementation. In this approach, the sample complexity is linearly dependent to the design space dimension and the sampling procedure allows to obtain offline an inner approximation of the chance-constrained set. This approach has been extended to a more generic setup in [10] and experimentally validated for the control of a spacecraft during rendezvous maneuvers. Beside the efficacy of the approach, the results highlighted the need to further reduce the computational load and the slowness of the proposed approach to comply with faster dynamics and low-cost, low-performance hardware.

Among challenging applications, the control of unmanned aerial vehicles (UAVs) during assorted scenarios, have been triggering the attention of MPC community. These platforms are typically characterized by fast dynamics and equipped with computationally-limited autopilots. In the last decade, different receding horizon techniques have been proposed, see e.g. [11], [12], [13], [14], including a stochastic approach by [15]. In this case, preliminary analysis have confirmed the effectiveness of the proposed offline sampling-based SMPC (OS-SMPC) strategy but the results highlighted also the need to further reduce the dimension of the optimization problem to comply with hardware requirements.

The main contribution of this paper is to propose a new methodology that combines the probabilistic-scaling approach proposed in [16], which allows to obtain a low-complexity inner approximation of the chance constrained set, with the SMPC approach of [9], [10]. In [16], authors

show how to scale a given set of manageable complexity around its center to obtain, with a user-defined probability, a region that is included in the chance constrained set. In this paper, we extend the aforementioned approach showing how it is possible to reduce the sample complexity via probabilistic scaling exploiting so-called simple approximating sets (SAS). The starting point consists in obtaining a first simple approximation of the "shape" of the probabilistic set. To design a candidate SAS, we propose two possibilities. The first one is based on the definition of an approximating set by drawing a fixed number of samples. On the other hand, the second case envisions the use of ℓ_p-norm based sets, first proposed in [17]. In particular, we consider as SAS a ℓ_1-norm *cross-polytope* and a ℓ_∞-norm *hyper-cube*. Solving a standard optimization problem, it is possible to obtain the center and the shape of the SAS, which will be later scaled to obtain the expected probabilistic guarantees following the approach described in [16]. Then, the scaled SAS is used in the classical SMPC algorithm to enforce constraints.

To validate the proposed approach, an agriculture scenario has been selected, because of the increasing interest of using drones in the agriculture 4.0 framework, as explained in [18], due to their great potential to support and address some of the most pressing dares in farming. And real-time quality data and crop monitoring are two of those challenges. In particular, UAVs could represent a favorable alternative to conventional farming machines, whenever clear advantages with respect to traditional methods, in terms of higher efficiency in operations, reduced environmental impact or enhanced human health and safety are sought. For this paper, the control objective envisions the proposed approach applied to a fixed-wing UAV performing a monitoring mission over a sloped vineyard, following a pre-defined snake path. The performance of the proposed approach in terms of tracking capabilities and computational load has been compared with those obtained exploiting the "classical" OS-SMPC scheme proposed in [15].

Notation: The set $\mathbb{N}_{>0}$ denotes the positive integers, the set $\mathbb{N}_{\geq 0} = \{0\} \cup \mathbb{N}_{>0}$ the non-negative integers, and $\mathbb{N}_a^b$ the integers interval $[a, b]$. Positive (semi)definite matrices A are denoted $A \succ 0$ $(A \succeq 0)$ and $\|x\|_A^2 \doteq x^T A x$. For vectors, $x \succeq 0$ $(x \preceq 0)$ is intended component-wise. Pr_a denotes the probabilistic distribution of a random variable a. Sequence of scalars/vectors are denoted with bold lower-case letters, i.e. $\mathbf{v}$.

II. OFFLINE SAMPLING-BASED STOCHASTIC MPC

In this section, we first recall the Stochastic MPC Framework proposed in [9], [10].

A. Problem setup

We consider the case of a discrete-time system subject to generic uncertainty $w_k \in \mathbb{R}^{n_w}$

$$x_{k+1} = A(w_k)x_k + B(w_k)u_k + a_w(w_k), \quad (1)$$

with state $x_k \in \mathbb{R}^n$, control input $u_k \in \mathbb{R}^m$, and the vector valued function $a_w(w_k)$ represent the additive disturbance

affecting the systems states. The system matrices $A(w_k)$ and $B(w_k)$, of appropriate dimensions, are (possibly nonlinear) functions of the uncertainty w_k at step k. The disturbances $(w_k)_{k \in \mathbb{N}_{\geq 0}}$ are modeled as realizations of the stochastic process $(W_k)_{k \in \mathbb{N}_{\geq 0}}$, on which take the following assumptions.

Assumption 1 (Random Disturbances): The disturbances W_k, for $k \in \mathbb{N}_{\geq 0}$, are independent and identically distributed (i.i.d.), zero-mean random variables with support $\mathbb{W} \subseteq \mathbb{R}^{n_w}$. Moreover, let $\mathbb{G} = \{(A(w_k), B(w_k), a_w(w_k)\}_{w_k \in \mathbb{W}}$, a polytopic outer approximation with N_c vertexes $\dot{\mathbb{G}} \doteq co\{A^j, B^j, a_w^j\}_{j \in \mathbb{N}_1^{N_c}} \supseteq \mathbb{G}$ exists and is known.

We can notice that the system can be augmented by a filter to model a specific stochastic processes of interest. The assumption of independent random variables in necessary to perform the offline computations discussed next while the need of a known outer bound is required to establish a safe operating region (see [9] for details). We remark that the system's representation in (1) is very general, and encompasses e.g. those in [9], [10], [19]. Given the model (1) and a realization of the state x_k at time k, state predictions l steps ahead are random variables, as well and are denoted $x_{l|k}$, to differentiate it from the realization x_{l+k}. Similarly $u_{l|k}$ denotes predicted inputs that are computed based on the realization of the state x_k.

The system is subject to p state and input chance constraints of the form[1]

$$\mathrm{Pr}_\mathbf{w} \left\{ [H_x]_j^T x_{l|k} + [H_u]_j^T u_{l|k} \leq 1 | x_k \right\} \geq 1 - \varepsilon_j,$$
$$l \in \mathbb{N}_{>0}, j \in \mathbb{N}_1^p, \quad (2)$$

with $\varepsilon_j \in (0, 1)$, and $H_x \in \mathbb{R}^{p \times n}$, $H_u \in \mathbb{R}^{p \times m}$, where $[H]_j^T$ denotes the j-th row of matrix H. The probability $\mathrm{Pr}_\mathbf{w}$ is measured with respect to the sequence $\mathbf{w} = \{w_i\}_{i > k}$. Hence, equation (2) states that the probability of violating the linear constraint $[H_x]_j^T x + [H_u]_j^T u \leq 1$ for any future realization of the disturbance should not be larger than ε_j.

The objective is to derive an asymptotically stabilizing control law for the system (1) such that, in closed loop, the constraints (2) are satisfied.

B. Stochastic Model Predictive Control

To solve the constrained control problem, a stochastic MPC algorithm is considered. The approach is based on repeatedly solving a stochastic optimal control problem over a finite, moving horizon, but implementing only the first control action. Defined the control sequence as $\mathbf{u}_k = (u_{0|k}, u_{1|k}, ..., u_{T-1|k})$, the prototype optimal control problem that is to be solved at each sampling time is given minimizing the cost function

$$J_T(x_k, \mathbf{u}_k) =$$
$$\mathbb{E} \left\{ \sum_{l=0}^{T-1} \left(x_{l|k}^\top Q x_{l|k} + u_{l|k}^\top R u_{l|k} \right) + x_{T|k}^\top P x_{T|k} \mid x_k \right\} \quad (3)$$

---

[1]The case where one wants to impose *hard* input constraints can be also be formulated in a similar framework, see e.g. [9].

with $Q \succ 0$, $R \succ 0$, and appropriately chosen $P \succ 0$, subject to the system dynamics (1) and constraints (2).

The online solution of the stochastic MPC problem remains a challenging task but several special cases, which can be evaluated exactly, as well as methods to approximate the general solution have been proposed in the literature. The approach followed in this work was first proposed in [9], [10], where an offline sampling scheme was introduced. Therein, with a prestabilizing input parametrization

$$u_{l|k} = Kx_{l|k} + v_{l|k}, \tag{4}$$

with suitably chosen control gain $K \in \mathbb{R}^{n \times m}$ and free optimization variables $v_{l|k} \in \mathbb{R}^m$, equation (1) is solved explicitly for the predicted states $x_{1|k}, \ldots, x_{T|k}$ and predicted inputs $u_{0|k}, \ldots, u_{T-1|k}$. In this case, the expected value of the finite-horizon cost (3) can be evaluated *offline*, leading to a quadratic cost function of the form

$$J_T(x_k, \mathbf{v}_k) = [x_k^T \ \mathbf{v}_k^T \ \mathbf{1}_n^T] \tilde{S} \begin{bmatrix} x_k \\ \mathbf{v}_k \\ \mathbf{1}_n \end{bmatrix} \tag{5}$$

in the deterministic variables $\mathbf{v}_k = (v_{0|k}, v_{1|k}, ..., v_{T-1|k})$ and x_k. The reader can refer to [10, Appendix A] for a detailed derivation of the cost matrix $\tilde{S}$.

Focusing now on the constraint definition, we can notice that by introducing the uncertainty sequence $\mathbf{w}_k = \{w_l\}_{l=k,\ldots,k+T-1}$, we can rewrite the j-th chance constraint defined by equation (2) as

$$\mathbb{X}_\varepsilon^j = \left\{ \begin{bmatrix} x_k \\ \mathbf{v}_k \end{bmatrix} \in \mathbb{R}^{n+mT} \mid \right.$$
$$\left. \mathsf{Pr}_{\mathbf{w}_k} \left\{ f_j^T(\mathbf{w}_k) \begin{bmatrix} x_k \\ \mathbf{v}_k \end{bmatrix} \leq 1 \right\} \geq 1 - \varepsilon \right\}, \tag{6}$$

with f_j being a function of the sequence of random variables $\mathbf{w}_k$. Again, the reader is referred to [10] for details on the derivation of f_j. The results in [9] show that, by exploiting results from statistical learning theory (cf. [20], [21]), we can construct an inner approximation $\underline{\mathbb{X}}^j$ of the constraint set $\mathbb{X}_\varepsilon^j$ by extracting N_{LT} i.i.d. samples $\mathbf{w}_k^{(i)}$ of $\mathbf{w}_k$ and taking the intersection of the sampled constraints, i.e.

$$\underline{\mathbb{X}}_{LT}^j = \left\{ \begin{bmatrix} x_k \\ \mathbf{v}_k \end{bmatrix} \in \mathbb{R}^{n+mT} \mid \right.$$
$$\left. f_j^T(\mathbf{w}_k^{(i)}) \begin{bmatrix} x_k \\ \mathbf{v}_k \end{bmatrix} \leq 1, \ i = 1, \ldots, N_{LT} \right\}, \tag{7}$$

In particular, it has been shown in [9], that for given probabilistic levels $\delta \in (0, 1)$ and $\varepsilon_j \in (0, 0.14)$, choosing the sample complexity

$$N_{LT}^j \geq \tilde{N}(n + mT, \varepsilon_j, \delta)$$
$$\doteq \frac{4.1}{\varepsilon} \left(\ln \frac{21.64}{\delta} + 4.39(n + mT) \log_2 \left(\frac{8e}{\varepsilon_j} \right) \right), \tag{8}$$

guarantees that with probability at least δ the sample approximation $\underline{\mathbb{X}}_{LT}^j$ is included in the original chance constraint $\mathbb{X}_\varepsilon^j$, i.e.

$$\mathsf{Pr} \left\{ \underline{\mathbb{X}}_{LT}^j \subseteq \mathbb{X}_\varepsilon^j \right\} \geq 1 - \delta, \quad j = 1, .., p. \tag{9}$$

Hence, exploiting these results, we obtain that the stochastic MPC problem can be well approximated by the following linearly constrained quadratic program

$$\min_{\mathbf{v}_k} \ J_T(x_k \mathbf{v}_k) \tag{10}$$
$$\text{s.t. } (x_k, \mathbf{v}_k) \in \underline{\mathbb{X}}_{LT}^j, \quad j = 1, .., p \tag{11}$$

While the result reduces the original stochastic optimization program to an efficiently solvable quadratic program, the ensuing number of constraints, equal to

$$N_{LT} = \sum_{i=1}^{T} N_{LT}^j,$$

may still be too large. For instance, even for a moderately sized MPC problem with $n = 5$ states, $m = 2$ inputs and horizon of $T = 10$, and for a reasonable choice of probabilistic $\varepsilon^j = 0.05$, $\delta = 10^{-6}$, we get $N_{LT}^j = 20,604$. For this reason, in [9] a post-processing analysis of the constraint set was proposed for removing redundant constraints. While it is indeed true that all the cumbersome computations may be performed offline, it is still the case that in applications with stringent requirements on the solution time the final number of inequalities may easily become unbearable. This observation motivates the approach presented in the next section, which builds upon the results presented in [16], showing how the probabilistic scaling approach leads to approximations of "controllable size," that can be directly used in applications.

III. COMPLEXITY REDUCTION VIA PROBABILISTIC SCALING

In this section, we consider the very general problem of finding a decision variable vector ξ, restricted to a set $\Xi \subseteq \mathbb{R}^{n_\xi}$, subject p uncertain linear inequalities. Formally, we consider uncertain inequalities of the form

$$F(q)\xi \leq g(q) \tag{12}$$

where $F(q) \in \mathbb{R}^{p \times n_\xi}$ and $g(q) \in \mathbb{R}^{n_p}$ are continuous function of the uncertainty vector $q \in \mathbb{R}^{n_q}$. The uncertainty vector q is assumed to be of random nature, with given probability distribution Pr_q and (possibly unbounded) support $\mathbb{Q}$. Hence, to each sample of q corresponds a different set of linear inequalities. We aim at finding an approximation of the ε-chance-constraint set, defined as

$$\mathbb{X}_\varepsilon \doteq \left\{ \xi \in \Xi \mid \mathsf{Pr}_q \left\{ F(q)\xi \leq g(q) \right\} \geq 1 - \varepsilon \right\} \tag{13}$$

that represents the region of the design space Ξ for which this probabilistic constraint is satisfied. Note that this captures exactly the SMPC setup discussed in the previous section. Indeed, the chance-constrained set in (6) is a special instance of (13), with $\xi = [x_k^T \ \mathbf{v}_k^T]^T$ and $q = \mathbf{w}_k$.

The characterization of the chance constrained set has several application in robust and stochastic control. A classical approach is to find inner convex approximation of the probabilistic set $\mathbb{X}_\varepsilon$, obtained for instance by means of applications of Chebyshev-like inequalities, see e.g. [22] and
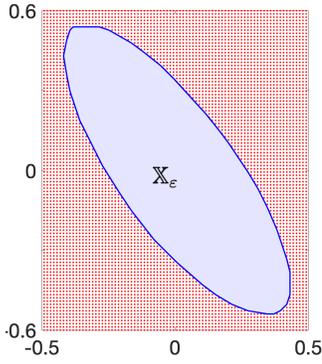
Fig. 1. Example of chance-constraint set for $\varepsilon = 0.05$, for 2D scalar linear constraints of the form $f(q)^T\xi \leq 1$, with $f(q) = q_1 \cdot q_2 \in \mathbb{R}^2$, $q_1 \in \mathbb{R}$ uniformly distributed in the interval $[0.5, 1.5]$ and $q_2 \in \mathbb{R}^2$ Gaussian with variance Σ. The set was obtained by evaluating the empirical probability via random sampling.

[23]. A recent approach, which is the one applied in the previous section to the SMPC problem, is instead based on the derivation of probabilistic approximations of the chance constraints set $\mathbb{X}_\varepsilon$ through sampling of the uncertainty. That is, we aim at constructing a set $\underline{\mathbb{X}}$ which is contained in $\mathbb{X}_\varepsilon$ *with high probability*.

Denote $F_j(q)$ and g_j the j-th row of $F(q)$ and j-th component of q respectively. Consider the binary functions

$$h_j(\xi, q) \doteq \left\{ \begin{array}{ll} 0 & \text{if } F_j(q)\xi \leq g_j(q) \\ 1 & \text{otherwise} \end{array} \right. , \quad j = 1, \ldots, p.$$

Now, if we define

$$h(\xi, q) \doteq \prod_{j=1}^{p} h_j(\xi, q).$$

we have that h is an $(1, p)$-boolean function since it can be expressed as a function of p boolean functions, each of them involving a polynomial of degree 1. See e.g. [21, Definition 7] for a precise definition of this sort of boolean functions. Suppose that we draw N i.i.d. samples $q^{(i)}$, $i = 1, \ldots, N$. Then, we can consider the (empirical) region $\mathbb{X}_N$ defined as

$$\mathbb{X}_N \doteq \{ \xi \in \mathbb{R}^{n_\xi} : h(\xi, q^{(i)}) = 0, i = 1, \ldots, N \}.$$

It has been proved in [21, Theorem 8], that if $\epsilon \in (0, 0.14)$ and N is chosen such that[2]

$$N \geq \frac{4.1}{\epsilon}\left(\ln \frac{21.64}{\delta} + 4.39n_\xi \log_2 \left(\frac{8ep}{\epsilon} \right) \right)$$

then $\mathbb{X}_N \subseteq \mathbb{X}_\varepsilon$ with a probability no smaller than $1 - \delta$.

We notice that $\mathbb{X}_N$ is a convex set, which is a desirable property in an optimization framework. However, the number of required samples N might be prohibitive for a real-time application. To tackle this issue, in this paper we exploit an appealing alternative approach proposed in [16], and we specialize it to the problem at hand. This work proposes a probabilistic scaling approach to obtain, with given confidence, an inner approximation of the chance constrained

---

set $\mathbb{X}_\varepsilon$ avoiding the computational burden due to the sample complexity raising in other strategies.

The main idea behind this approach consist in first obtaining a simple initial approximation of the "shape" of the probabilistic set $\mathbb{X}_\varepsilon$ by exploiting simple approximating sets of the form

$$x_c \oplus \underline{\mathbb{S}}.$$

This set is not required to have *any* guarantees of probabilistic nature. Instead, to derive such probabilistic guaranteed set, a scaling procedure is devised. In particular, an optimal scaling factor γ is derived so that the set scaled around its center x_c

$$\underline{\mathbb{S}}(\gamma) \doteq x_c \oplus \gamma\underline{\mathbb{S}}. \tag{14}$$

is guaranteed to be an inner approximation of $\mathbb{X}_\varepsilon$ with the desired confidence level δ.

A. Simple Approximating Sets

The idea at the basis of the proposed approach is to define Simple Approximating Sets (SAS), which represent specifically defined sets with a low – and pre-defined – number of constraints. First, we note that the most straightforward way to design a candidate SAS is to draw a fixed number N_S of uncertainty samples, and to construct a sampled approximation as follows:

1. Sampled-poly

$$\underline{\mathbb{S}}_S = \bigcap_{i=1}^{N_S} \mathbb{X}_i \tag{15}$$

where

$$\mathbb{X}_i \doteq \left\{ \xi \in \Xi \mid F(q^{(i)}) \leq g(q^{(i)}), \quad i = 1, \ldots, N_S \right\} \tag{16}$$

Clearly, if $N_S << N_{LT}$, the probabilistic properties of $\underline{\mathbb{S}}_S$ before scaling will be very bad. However, at this point we do not care, since the probabilistic scaling proposed in Section III-B will take care of this.

A second way to construct a SAS considered in this paper exploits a class of ℓ_p-norm based sets introduced in [17] as follows

$$\mathcal{A}(x_c, P) \doteq \{\xi \in \mathbb{R}^{n_\xi} \mid \xi = x_c + Pz, z \in \mathcal{B}_p\}, \tag{17}$$

where $\mathcal{B}_p \subset \mathbb{R}^{n_\xi}$ is the unit ball in the p norm, x_c is the center and $P = P^T \succeq 0$ is the so-called *shape* matrix. In particular, we note that for $p = 1, \infty$ these sets take the form of polytopes with fixed number of facets/vertices. Hence, we introduce the following two SAS:

2. ℓ_1-poly

$$\underline{\mathbb{S}}_1 = \{\xi \in \mathbb{R}^{n_\xi} \mid \xi = x_c + Pz, \ \|z\|_1 \leq 1\}, \tag{18}$$

defined starting from a *cross-polytope*, also known as *diamond*, of order n_ξ with $2n_\xi$ vertices and 2^{n_ξ} facets.

3. ℓ_∞-poly

$$\underline{\mathbb{S}}_\infty = \{\xi \in \mathbb{R}^{n_\xi} \mid \xi = x_c + Pz, \ \|z\|_\infty \leq 1\}, \tag{19}$$

---

[2]Note the difference under the $\log_2$ with respect to (8).

defined starting from a *hyper-cube* of dimension n_ξ with 2^{n_ξ} vertices and $2n_\xi$ facets.

Hence, the problem becomes designing the center and shape parameters (x_c, P) of the set $\mathbb{S}_1$ (resp. $\mathbb{S}_\infty$) so that they represent in the best possible way the set $\mathbb{X}_\varepsilon$. To this end, we start from a *sampled design polytope*

$$\mathbb{D} = \bigcap_{i=1}^{N_D} \mathbb{X}_i,$$

with a fixed number of samples N_D, and construct the largest set $\mathbb{S}_1$ (resp. $\mathbb{S}_\infty$) contained in $\mathbb{D}$. It is easily observed that to obtain the largest ℓ_1-poly inscribed in $\mathbb{D}$, we need to solve the following convex optimization problem

$$\max_{x_c, C} \quad \text{tr}(P) \tag{20}$$
$$\text{s.t.} \quad P \succeq 0,$$
$$f_i^T P z^{[j]} \leq g_i - f_i^T x_c$$
$$i = 1, \dots, N_D, \quad z^{[j]} \in \mathcal{V}_1, \tag{21}$$

where $\mathcal{V}_1 = \{z^{[1]}, \dots, z^{[2n_\xi]}\}$ are the vertices of the unit cross-polytope while the vertices of the optimal ℓ_1-poly can then be obtained as

$$\xi^{[j]} = x_c + P z^{[j]}, \quad j = 1, \dots, 2n_\xi. \tag{22}$$

It should be remarked that, from these vertices, one could then recover the corresponding 2^{n_ξ} linear inequalities, each one defining a facet of the rotated diamond. However, this procedure, besides being computationally extremely demanding (going from a vertex-description to a linear inequality description of a polytope is known to be NP hard, [24]), would lead to an exponential number of linear inequalities, thus rendering the whole approach not viable. Instead, we exploit the following equivalent formulation of (17), see e.g. [17] for details

$$\mathbb{S}_1 = \{\xi \in \mathbb{R}^{n_\xi} \mid \|Mx - c\|_1 \leq 1\} \tag{23}$$

where $M \doteq P^{-1}$ and $c \doteq P^{-1} x_c$. From a computational viewpoint, this second approach results to be more appealing. Indeed, using a slack variable ζ, it is possible to obtain the following system of $3n_\xi + 1$ linear inequalities

$$\begin{cases} m_i^T \xi - c_i \leq \zeta_i, & i = 1, \dots, n_\xi \\ -m_i^T \xi + c_i \leq \zeta_i, & i = 1, \dots, n_\xi \\ \zeta_i \geq 0, & i = 1, \dots, n_\xi \\ \sum_i^{n_\xi} \zeta_i \leq 1, \end{cases}$$

The same convex optimization problem of (20) could be solved to define the center and the shape of the *largest* ℓ_∞-poly inscribed in $\mathbb{D}$. However, this would involve an exponential number of vertices 2^{n_ξ}. To avoid this, an approach based on Farkas lemma can be adopted, exploiting again a formulation in terms of linear inequalities. The details are not reported here due to space limitations. In this second case, obtained the center x_c and the rotation matrix P, the corresponding $\mathcal{H}$-poly has only $2n_\xi$ hyper-planes, each one representing a different linear inequality.

Once the initial SAS, $\mathbb{S}_S$ and the ℓ_1- and ℓ_∞-polys, i.e. $\mathbb{S}_1$ and $\mathbb{S}_\infty$ respectively, has been evaluated in terms of linear inequalities, the probabilistic scaling approach can be applied to determine the corresponding scaling factor γ. The scaling procedure is described in details in the paper [16]. For the sake of completeness, in the next subsection we recall its basic ideas and illustrate its application to the SAS case.

B. SAS probabilistic scaling

Given a candidate SAS set, the following simple algorithm can be used to guarantee with prescribed probability $1 - \delta$ that the scaled set $\mathbb{S}(\gamma)$ is a good inner approximation of $\mathbb{X}_\varepsilon$.

Algorithm 1 Probabilistic SAS Scaling

1: Given probability levels ε and δ, let

$$N_\gamma \geq \frac{7.67}{\varepsilon} \ln \frac{1}{\delta} \quad \text{and} \quad r = \left\lceil \frac{\varepsilon N_\gamma}{2} \right\rceil.$$

2: Draw N_γ samples of the uncertainty $q^{(1)}, \dots, q^{(N_\gamma)}$
3: **for** $i = 1$ to N_γ **do**
4:      Solve the optimization problem

$$\gamma_i \doteq \arg \max \gamma \tag{24}$$
$$\text{s.t.} \quad \mathbb{S}(\gamma) \subseteq \mathbb{X}_i$$

5: **end for**
6: Return the r-th smallest value of γ_i.

A few comments are at hand regarding the algorithm above. In step 4, for each uncertainty sample $q^{(i)}$ one has to solve a convex optimization problem, which amounts at finding the largest value of γ such that $\mathbb{S}(\gamma)$ is contained in the set $\mathbb{X}_i$ defined in (16). Then, in step 6, one has to reorder the set $\{\gamma_1, \gamma_2, \dots, \gamma_{N_\gamma}\}$ so that the first element is the smallest one, the second element is the second smallest one, and so on and so fort, and then return the r-th element of the reordered sequence. The following Lemma applies to Algorithm 1.

Lemma 1: Given a candidate SAS set in the form $\mathbb{S}(\gamma) = x_c \oplus \gamma \mathbb{S}$, assume that $x_c \in \mathbb{X}_\varepsilon$. Then, Algorithm 1 guarantees that

$$\mathbb{S}(\gamma) \subseteq \mathbb{X}_\varepsilon$$

with probability at least $1 - \delta$.

Proof to Lemma 1 is reported in Appendix.

C. Illustrating Example

To better illustrate the proposed approach, and to highlight its main features, we first consider a simple three-dimensional examples ($n_\xi = 3$), with scalar uncertain linear inequalities of the form

$$f(q)^T \xi \leq 1$$

with $f(q) = q_1 q_2$, with $q_1 \in \mathbb{R}$ uniformly distributed in the interval $[0.5, 1.5]$ and $q_2 \in \mathbb{R}^3$ zero-mean Gaussian distribution. Note that, for $n_\xi = 3$, the ℓ_1 and ℓ_∞-polys have
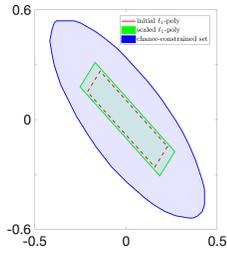
Fig. 2. Visualization of the probabilistic scaling procedure for the 2D chance-constraint set considered in Figure III-B. The initial set (in red) was obtained by constructing the largest $\mathbb{S}_1$-poly enclosed in the set $\mathbb{S}$.

10 and 6 facets, respectively, irrespective to the number of design samples N_D used to preliminary obtain the generic polyhedron $\mathbb{D}$. However, as we will see, the number of constraints N_D employed to design the initial SAS plays a significant role in the final outcome of the procedure. To show this, we performed two different tests, where the number of design samples was set to $N_D = 100$ and $N_D = 1,000$. The results are shown in Figures 3 and 4 respectively, for both the ℓ_1 (left) and ℓ_∞ (right) cases. Algorithm 1 was applied in all cases with $\varepsilon = 0.05$ and $\delta = 10^{-6}$, leading to $N_\gamma = 2,063$ and $r = 103$. For allowing a better comparison, the same set of samples where considered for the evaluation of the scaling factor in all examples. These samples lead to N_γ random hyper-planes which define a polyhedron represented (in black) in the figures.
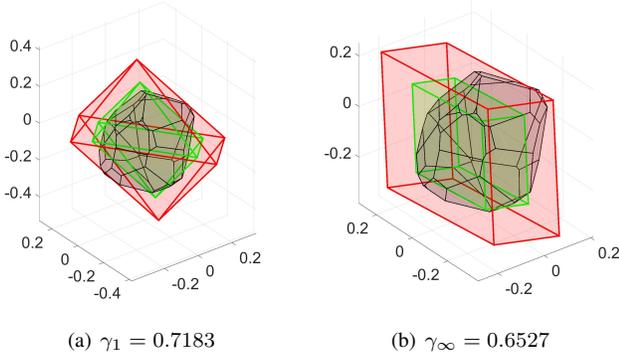

(a) $\gamma_1 = 0.7183$

(b) $\gamma_\infty = 0.6527$

Fig. 3. Scaled ℓ_1-poly (a) and ℓ_∞-poly (b) obtained starting from $N_D = 100$.

It can be observed that when N_D is small, the ensuing initial ℓ_1- (resp. ℓ_∞-) poly is large, and Algorithm 1 returns a scaling factor γ which is less than one (Fig. 3). Hence, the probabilistic scaling produces a "deflation" of the original set so to guarantee the probabilistic constraints. Vice-versa, for large N_D (Fig. 4), the scaling produces an inflation, returning a value of γ larger than one.

Finally, we compared the ℓ_1- / ℓ_∞- polys with the naive approach based on sampled polytope $\mathbb{S}_S$. Notice that, to allow a fair comparison, we should select a number of hyper-planes comparable with the number of linear inequalities defining $\mathbb{S}_1$ and $\mathbb{S}_\infty$. In Fig. 5(a), we represent the initial
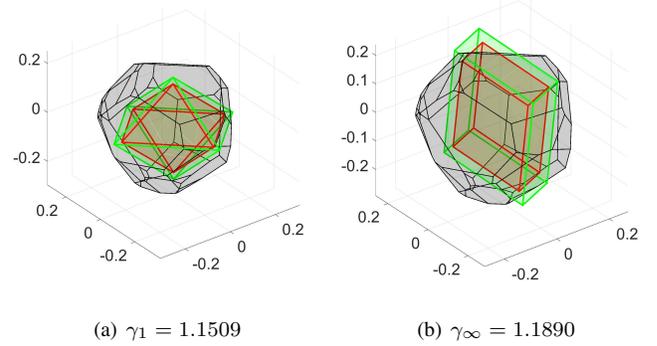

(a) $\gamma_1 = 1.1509$

(b) $\gamma_\infty = 1.1890$

Fig. 4. Scaled ℓ_1-poly (a) and ℓ_∞-poly (b) obtained starting from $N_D = 1,000$.

and final polytopes. Then, we also generated two additional sampled-polys with $N_S = 100$ and $N_S = 1,000$, i.e. equal to the number of hyper-planes used to generate the design polyhedrons $\mathbb{D}$ for the previous case. These are depicted in

(a) $N_S = 10$, $\gamma_S = 0.1754$

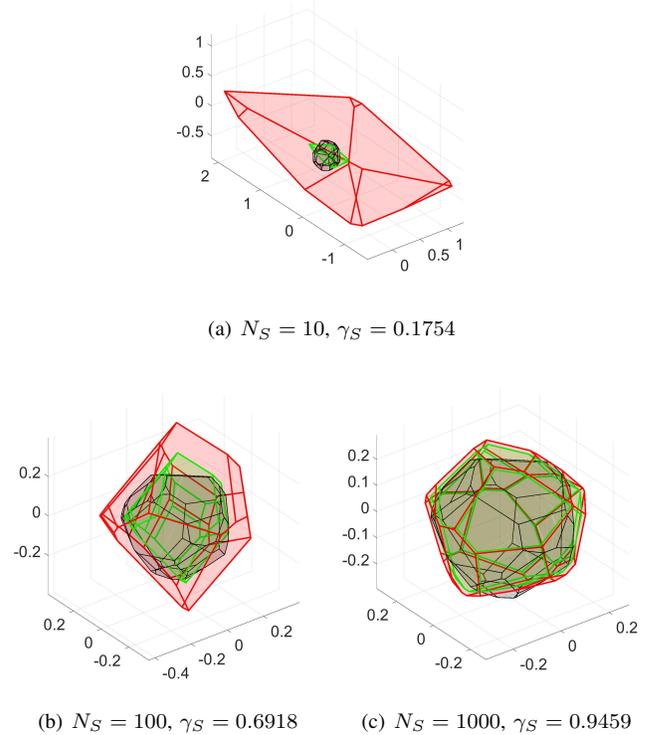(b) $N_S = 100$, $\gamma_S = 0.6918$

(c) $N_S = 1000$, $\gamma_S = 0.9459$

Fig. 5. Scaled sampled-polys obtained starting from $N_S = 10$ (a), $N_S = 100$ (b) and $N_S = 1,000$ (c) and for $N_\gamma = 2,063$.

Figs. 5(b)-5(c) while the volumes of the different SASs are reported in Table I.

IV. UAV CONTROL OVER A SLOPED VINEYARD

The selected application involves a fixed-wing UAV performing a monitoring mission over a Dolcetto vineyard at Carpeneto, Alessandria, Italy ($44°40'55.6''$N, $8°37'28.1''$E). The Mission Planner of ArduPilot open source autopilot has

TABLE I
VOLUME OF THE DIFFERENT SASs CONSIDERED IN EXAMPLE 1.

$\mathbb{S}$	N_S	N_D	V
$\mathbb{S}_{10}$	10	—	0.0091
$\mathbb{S}_{100}$	100	—	0.0403
$\mathbb{S}_{1000}$	1000	—	0.0526
$\mathbb{S}_1$	—	100	0.0176
$\mathbb{S}_1$	—	1000	0.0258
$\mathbb{S}_\infty$	—	100	0.0131
$\mathbb{S}_\infty$	—	1000	0.0175

been used to identify a grid pattern with a peculiar path orientation with respect to the grapevine rows, as shown in Fig. 6. The main objective is to provide proper control


Fig. 6.   Carpeneto vineyard, Piedmont, Italy (credit: Google).

capabilities to a fixed-wing UAV to guarantee a fixed relative altitude with respect to the terrain of 150 m while following the desired optimal path defined by the guidance algorithm (described in detail in [25]), maintaining a constant airspeed, i.e. $V_{ref} = 12$ m/s. The controllability of the aircraft shall be guaranteed despite the presence of external disturbance due to a fixed-direction wind turbulence, which intensity can randomly vary among ± 1 m/s.

For validation purpose, the longitudinal control of the UAV has been provided exploiting both OS-SMPC and the new PS-SMPC approach. In this case study, we have that the state variable are the longitudinal component of the total airspeed in body axes u, the angle of attack α, the pitch angle θ, the pitch rate q, and the altitude h. On the other hand, the control variables are represented by the throttle command ΔT and the elevator deflection δ_e. Hence, we have $n = 5$ and $m = 2$ while the prediction horizon T has been set equal to 15. Consequently, setting $\varepsilon = 0.05$, $\delta = 10^{-6}$, we get $N_{LT} = 20,604$ and $N_\gamma = 2,063$. On the other hand, the sample complexity selected for generating the ℓ_1-poly has been set equal to $N_D = 100$ obtaining $(n + mT) \cdot N_D = 3,500$ hyper-planes but only $3(n + mT) + 1 = 107$ linear constraints implemented online.

The preliminary results are represented in Fig. 7 as 3D trajectories and in Fig. 8 as controlled states with respect to reference signals. We can notice that both MPC schemes provide acceptable tracking capabilities, despite larger (but
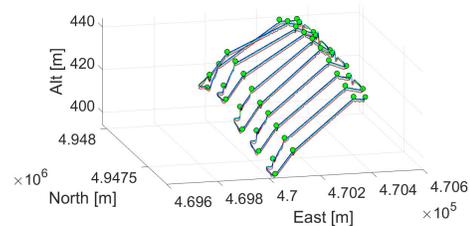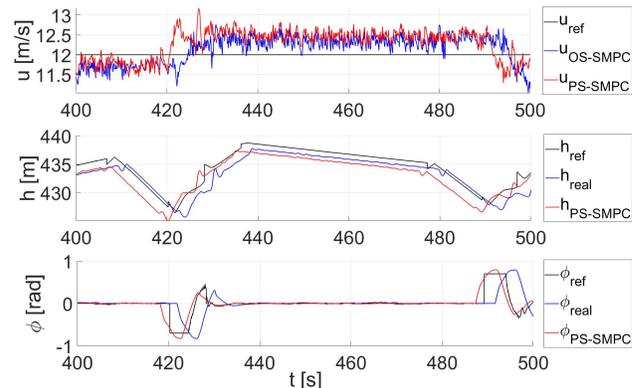

(a) OS-SMPC
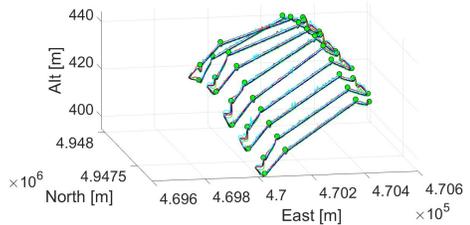
(b) PS-SMPC

Fig. 7.   UAV controlled trajectories obtained running OS-SMPC (a) and PS-SMPC (b) five times each.

Fig. 8.   Zoom-in on the behavior of controlled state variables, i.e. airspeed u, altitude h and roll angle ϕ, obtained exploiting OS-SMPC (blue lines) and PS-SMPC (red lines) with respect to corresponding reference signals (black lines), i.e. u_{ref}, h_{ref} and ϕ_{ref}.

still acceptable) oscillations can be observed in 7(b) when the scaled set is exploited. More interesting results are reported in Tab. II in terms of maximum and average values of the computational time required to solve *online* the finite-horizon optimal control problem, evaluated for 5 different run each. The results show a significant reduction (about 100 times lower) of the computational load when a lower complexity constraint set is employed. This makes the stochastic MPC approach not only effective from a performance viewpoint but also presumably compliant with the computational constraint coming from autopilot hardware.

V. CONCLUSIONS

In this paper, we proposed a novel approach which exploits a probabilistic scaling technique recently proposed by some of the authors to derive a novel Stochastic MPC scheme. The

TABLE II

MAXIMUM AND AVERAGE COMPUTATIONAL COST REQUIRED BY OS-SMPC AND PS-SMPC APPROACHES FOR ONLINE SOLVING THE OPTIMIZATION PROBLEM DURING EACH RUN.

n.	$t_{c_{MAX_{OS}}}$	$t_{c_{AVG_{OS}}}$	$t_{c_{MAX_{PS}}}$	$t_{c_{AVG_{PS}}}$
1	2.0959	0.4178	0.0966	0.0087
2	0.5394	0.3291	0.0215	0.0088
3	5.1215	0.4546	0.1065	0.0045
4	2.1497	0.5434	0.2628	0.0086
5	2.9411	0.5626	0.7221	0.0190

introduced framework exhibits a lower computational complexity, while sharing the appealing probabilistic guarantees of off-line sampling.

APPENDIX

The proof of to Lemma 1 follows from Proposition 1 in [16], which guarantees that, for given $r \geq 0$, $\Pr\{\mathbb{S}(\gamma) \subseteq \mathbb{X}_\varepsilon\}$ is guaranteed if the scaling is performed on a number of samples such that

$$N \geq \frac{1}{\varepsilon}\left(r - 1 + \ln\frac{1}{\delta} + \sqrt{2(r-1)\ln\frac{1}{\delta}}\right). \qquad (25)$$

Since $r = \lceil \frac{\varepsilon N}{2}\rceil$, we have that $r - 1 \leq \frac{\varepsilon N}{2}$. Thus, inequality (25) is satisfied if

$$
\begin{aligned}
N &\geq \frac{1}{\varepsilon}\left(\frac{\varepsilon N}{2} + \ln\frac{1}{\delta} + \sqrt{\varepsilon N \ln\frac{1}{\delta}}\right) \\
&= \frac{N}{2} + \frac{1}{\varepsilon}\ln\frac{1}{\delta} + \sqrt{N\frac{1}{\varepsilon}\ln\frac{1}{\delta}}.
\end{aligned}
$$

Letting[3] $\nabla \doteq \sqrt{N}$ and $\alpha \doteq \sqrt{\frac{1}{\varepsilon}\ln\frac{1}{\delta}}$, the above inequality rewrites $\nabla^2 - 2\alpha\nabla - 2\alpha^2 \geq 0$, which has unique positive solution $\nabla \geq (1 + \sqrt{3})\alpha$, which rewrites as $N \geq \frac{(1+\sqrt{3})^2}{\varepsilon}\ln\frac{1}{\delta}$.

The formula in Algorithm 1 follows by observing that $(1 + \sqrt{3})^2 < 7.67$.

REFERENCES

[1] M. Farina, L. Giulioni, and R. Scattolini, "Stochastic linear model predictive control with chance constraints–a review," *Journal of Process Control*, vol. 44, pp. 53–67, 2016.

[2] R. Tempo, G. Calafiore, and F. Dabbene, *Randomized algorithms for analysis and control of uncertain systems: with applications*. Springer Science & Business Media, 2012.

[3] G. C. Calafiore and M. C. Campi, "The scenario approach to robust control design," *IEEE Transactions on Automatic Control*, vol. 51, no. 5, pp. 742–753, 2006.

[4] G. Schildbach, L. Fagiano, C. Frei, and M. Morari, "The scenario approach for stochastic model predictive control with bounds on closed-loop constraint violations," *Automatica*, vol. 50, no. 12, pp. 3009–3018, 2014.

[5] J. M. Grosso, P. Velarde, C. Ocampo-Martinez, J. M. Maestre, and V. Puig, "Stochastic model predictive control approaches applied to drinking water networks," *Optimal Control Applications and Methods*, vol. 38, no. 4, pp. 541–558, 2017.

[6] H. A. Nasir, A. Carè, and E. Weyer, "A randomised approach to flood control using value-at-risk," in *2015 54th IEEE Conference on Decision and Control (CDC)*. IEEE, 2015, pp. 3939–3944.

[7] D. Van Hessem and O. Bosgra, "Stochastic closed-loop model predictive control of continuous nonlinear chemical processes," *Journal of Process Control*, vol. 16, no. 3, pp. 225–241, 2006.

[8] R. M. Vignali, F. Borghesan, L. Piroddi, M. Strelec, and M. Prandini, "Energy management of a building cooling system with thermal storage: An approximate dynamic programming solution," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 2, pp. 619–633, 2017.

[9] M. Lorenzen, F. Dabbene, R. Tempo, and F. Allgöwer, "Stochastic MPC with offline uncertainty sampling," *Automatica*, vol. 81, no. 1, pp. 176–183, 2017.

[10] M. Mammarella, M. Lorenzen, E. Capello, H. Park, F. Dabbene, G. Guglieri, M. Romano, and F. Allgöwer, "An offline-sampling SMPC framework with application to autonomous space maneuvers," *IEEE Transactions on Control Systems Technology*, pp. 1–15, 2018.

[11] M. Kamel, T. Stastny, K. Alexis, and R. Siegwart, "Model predictive control for trajectory tracking of unmanned aerial vehicles using robot operating system," in *Robot Operating System (ROS)*. Springer, 2017, pp. 3–39.

[12] K. Alexis, C. Papachristos, R. Siegwart, and A. Tzes, "Robust model predictive flight control of unmanned rotorcrafts," *Journal of Intelligent & Robotic Systems*, vol. 81, no. 3-4, pp. 443–469, 2016.

[13] T. J. Stastny, A. Dash, and R. Siegwart, "Nonlinear mpc for fixed-wing UAV trajectory tracking: Implementation and flight experiments," in *AIAA Guidance, Navigation, and Control Conference*, 2017, p. 1512.

[14] N. Michel, S. Bertrand, G. Valmorbida, S. Olaru, and D. Dumur, "Design and parameter tuning of a robust model predictive controller for UAVs," in *2017 20th IFAC World Congress*, 2017.

[15] M. Mammarella, E. Capello, F. Dabbene, and G. Guglieri, "Sample-based SMPC for tracking control of fixed-wing UAV," *IEEE Control Systems Letters*, vol. 2, no. 4, pp. 611–616, 2018.

[16] T. Alamo, V. Mirasierra, F. Dabbene, and M. Lorenzen, "Safe approximations of chance constrained sets by probabilistic scaling," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 1380–1385.

[17] F. Dabbene, C. Lagoa, and P. Shcherbakov, "On the complexity of randomized approximations of nonconvex sets," in *2010 IEEE International Symposium on Computer-Aided Control System Design*. IEEE, 2010, pp. 1564–1569.

[18] G. Sylvester, G. Rambaldi, D. Guerin, A. Wisniewski, N. Khan, J. Veale, and M. Xiao, "E-agriculture in action-drones for agriculture. food and agriculture organization of the united nations and international telecommunication union," *Bangkok: Food and Agriculture Organization of the United Nations and International Telecommunication Union. Retrieved July*, vol. 19, 2018.

[19] M. Lorenzen, F. Dabbene, R. Tempo, and F. Allgöwer, "Constraint-tightening and stability in stochastic model predictive control," *IEEE Transactions on Automatic Control*, vol. 62, no. 7, pp. 3165–3177, 2017.

[20] M. Vidyasagar, *Learning and Generalisation: with Applications to Neural Networks*. Springer Science & Business Media, 2013.

[21] T. Alamo, R. Tempo, and E. F. Camacho, "Randomized strategies for probabilistic solutions of uncertain feasibility and optimization problems," *IEEE Transactions on Automatic Control*, vol. 54, no. 11, pp. 2545–2559, 2009.

[22] S. Yan, P. Goulart, and M. Cannon, "Stochastic model predictive control with discounted probabilistic constraints," in *2018 European Control Conference (ECC)*. IEEE, 2018, pp. 1003–1008.

[23] L. Hewing and M. N. Zeilinger, "Stochastic model predictive control for linear systems using probabilistic reachable sets," in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, pp. 5182–5188.

[24] V. Kaibel and M. E. Pfetsch, "Some algorithmic problems in polytope theory," in *Algebra, geometry and software systems*. Springer, 2003, pp. 23–47.

[25] M. Mammarella, G. Ristorto, E. Capello, N. Bloise, G. Guglieri, and F. Dabbene, "Waypoint tracking via tube-based robust model predictive control for crop monitoring with fixed-wing UAVs," in *2019 IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor)*. IEEE, 2019, pp. 19–24.

---

[3]Note that both quantities under square root are positive.