Reinforcement Learning based on MPC and the Stochastic Policy Gradient Method

Sébastien Gros, Mario Zanon

Abstract— **In this paper, we present a methodology to implement the stochastic policy gradient method using actor-critic techniques, when the policy is approximated using an MPC scheme. The paper proposes a computationally inexpensive approach to build a stochastic policy generating samples that are guaranteed to be feasible for the MPC constraints. For a continuous input space, imposing hard constraints on the policy poses technical difficulties in the computation of the score function of the policy, required in the policy gradient computation. We propose an approach that solves this issue, and detail how the score function can be computed based on parametric Nonlinear Programming and primal-dual interior point. The approach is illustrated on a simple example.**

I. Introduction

Reinforcement Learning (RL) offers useful tools for tackling Markov Decision Processes (MDP) without relying on a detailed model of the probability distributions underlying the state transitions [3], [21]. RL has drawn attention thanks to its relatable accomplishments, e.g., making it possible for robots to learn to walk or fly without supervision [1], [22]. In the recent RL literature, unstructured function approximation techniques are often used to carry the policy approximation needed in RL, such as, e.g., Deep Neural Networks (DNN). Structured function approximations based on formal control methods have recently gained the attention of the community [2], [7], [9], [11], [13], [14], [16], [18], [20], with the aim of providing formal closed-loop guarantees, and making it possible to directly use knowledge of the system.

In that context, Model Predictive Control (MPC) is an interesting candidate to carry the policy. Indeed, provided that a (possibly inaccurate) model of the real system is available, MPC delivers a suboptimal but typically reasonable policy for the real system. Moreover, MPC can treat explicitly hard constraints, which are typically used to impose limitations on the evolution of the state and inputs of the system. Robust MPC techniques can be used to impose safety guarantees on the closed-loop behavior of the real system under the MPC-based policy. Finally, because it seeks to minimize a given cost and respect constraints explicitly, the behavior of an MPC-based policy is often easier to interpret than the one of a generic policy approximation.

The use of MPC as a function approximator for RL is formally justified and detailed in [7], where it is shown that—under some stability assumptions—MPC schemes can theoretically generate jointly the optimal value function, action value function and policy underlying an MDP even though the model does not capture the real system correctly. This can be achieved via modifications of the MPC cost and constraints. This approach has been used in [8], [9], [17], [23], [24], [25] to propose adaptations of MPC schemes using RL techniques, allowing one to improve the closed-loop performance using data collected on the real system.

In [6], [10], [15] supervised learning is used to approximate an MPC feedback law, under the assumption that the nominal model used in MPC is exact, i.e., it describes the real system with no prediction error. In this paper, we assume that the MPC model can represent the system dynamics only approximately, and we perform a data-based adaptation of the MPC parameters to optimize its closed-loop performance.

Among the well-established RL methods, policy gradient methods based on Actor-Critic (AC) techniques [21] offer an attractive approach because arguably, unlike the Q-learning method, they are based on genuine conditions of optimality of the closed-loop policy. The stochastic policy gradient method is a popular AC approach, and rather simple assumptions are required for the method to work, making it fairly robust to use. In this paper, we investigate the use of the stochastic policy gradient approach in the context of MPC-based policies, and, more generally, constrained policies.

If constraints are to be handled, one expects the stochastic policy to generate actions that respect these constraints, i.e., that are feasible for the MPC scheme. An approach to do that, discussed in [9], consists in projecting the policy (supported by any function approximator) onto the set of actions that are feasible for MPC. In this paper instead, we directly construct the policy based on MPC such that it is feasible by construction. We first discuss how this requirement can be handled. We then discuss the technical issues resulting from having a stochastic policy limited by constraints, and propose an approach to circumvent them. We finally detail how the score function of the stochastic policy—central to computing the policy gradient—can be computed.

The paper is organized as follows. Section II provides background material on RL. Section III presents the MPC-based policy approximation, and details how to build a feasible stochastic policy from an MPC scheme. Section IV discusses how the score function—central to the policy gradient evaluation—can be computed. Section V provides further algorithmic details, and Section VI provides an academic example and Section VII provides conclusions.

II. Background

In the following, we will consider that the dynamics of the real system are described as a stochastic process on continuous state-input spaces, which can be modeled

Sébastien Gros is with the Department of Cybernetic, NTNU, Norway. Mario Zanon is with the IMT School for Advanced Studies Lucca, Italy.

only inaccurately. We will furthermore consider stochastic policies $\pi_{\boldsymbol{\theta}}$, parametrized by parameter $\boldsymbol{\theta}$, taking the form of probability densities:

$$\pi_{\boldsymbol{\theta}}\left[\mathbf{a}\,|\,\mathbf{s}\right] : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}_+, \tag{1}$$

denoting the probability density of selecting input $\mathbf{a}$ when the system is in state $\mathbf{s}$. For a given stage cost function $L(\mathbf{s},\mathbf{a}) \in \mathbb{R}$ and a discount factor $\gamma \in [0,1]$, the performance of a policy $\pi_{\boldsymbol{\theta}}$ is assessed via the total expected discounted cost:

$$J(\pi_{\boldsymbol{\theta}}) = \mathbb{E}_{\pi_{\boldsymbol{\theta}}}\left[\sum_{k=0}^{\infty} \gamma^k L(\mathbf{s}_k,\mathbf{a}_k) \,\middle|\, \mathbf{a}_k \sim \pi_{\boldsymbol{\theta}}\left[\,.\,|\,\mathbf{s}_k\right]\right]. \tag{2}$$

The optimal policy parameters $\boldsymbol{\theta}_\star$ are then given by:

$$\boldsymbol{\theta}_\star = \arg\min_{\boldsymbol{\theta}} J(\pi_{\boldsymbol{\theta}}). \tag{3}$$

The policy gradient $\nabla_{\boldsymbol{\theta}} J(\pi_{\boldsymbol{\theta}})$ associated with the stochastic policy $\pi_{\boldsymbol{\theta}}$ is instrumental in finding $\boldsymbol{\theta}_\star$, and can be obtained using various actor-critic methods [21]. In this paper, we will use the actor-critic formulation:

$$\nabla_\theta J(\pi_{\boldsymbol{\theta}}) = \mathbb{E}_{\boldsymbol{\pi}_{\boldsymbol{\theta}}}\left[\nabla_\theta \log \pi_{\boldsymbol{\theta}}\, \delta_{\pi_{\boldsymbol{\theta}}}\right], \tag{4}$$

$$\delta_{\pi_{\boldsymbol{\theta}}} := L\left(\mathbf{s},\mathbf{a}\right) + \gamma V_{\pi_{\boldsymbol{\theta}}}\left(\mathbf{s}_+\right) - V_{\pi_{\boldsymbol{\theta}}}\left(\mathbf{s}\right), \tag{5}$$

where the value function $V_{\pi_{\boldsymbol{\theta}}}$, associated with a given policy $\pi_{\boldsymbol{\theta}}$ is typically approximated and evaluated via Temporal-Difference (TD) techniques [21]. RL requires exploration to compute the policy gradient, which can be generated via perturbations over a deterministic policy. In continuous input spaces, it is common to define the stochastic policy $\pi_{\boldsymbol{\theta}}$ as an arbitrary density centred at a deterministic policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}$.

The advantage of policy gradient methods over a direct exploration of the policy parameters $\boldsymbol{\theta}$ is that open-ended scenarios can be handled and the directions obtained from the policy gradient tend to be less noisy than those obtained from exploring the policy parameters directly. Policy gradient methods can however become noisy if the approximation of the value function $V_{\pi_{\boldsymbol{\theta}}}$ is poor [21].

III. MPC-Based Policy Approximation

In this paper, we will build a stochastic policy $\pi_{\boldsymbol{\theta}}$ based on a perturbation of a deterministic policy $\boldsymbol{\pi}_\theta$. That deterministic policy will be based on a parametric Model Predictive Control (MPC) scheme. More specifically, for a given state $\mathbf{s}$ of the real system, the MPC scheme generates a deterministic policy given by:

$$\boldsymbol{\pi}_\theta\left(\mathbf{s}\right) = \mathbf{u}_0^\star\left(\mathbf{s},\boldsymbol{\theta}\right) \in \mathbb{R}^m, \tag{6}$$

where $\mathbf{u}_0^\star$ is the first element of the input sequence $\mathbf{u}^\star = \left\{\mathbf{u}_0^\star,\ldots,\mathbf{u}_{N-1}^\star\right\}$ resulting from solving the MPC scheme:

$$\mathbf{u}^\star(\mathbf{s},\boldsymbol{\theta}),\mathbf{x}^\star(\mathbf{s},\boldsymbol{\theta}) = \arg\min_{\mathbf{u},\mathbf{x}}\; T_{\boldsymbol{\theta}}(\mathbf{x}_N) + \sum_{k=0}^{N-1} \ell_{\boldsymbol{\theta}}(\mathbf{x}_k,\mathbf{u}_k) \tag{7a}$$

$$\text{s.t.}\;\; \mathbf{x}_{k+1} = \mathbf{f}_{\boldsymbol{\theta}}\left(\mathbf{x}_k,\mathbf{u}_k\right),\;\; \mathbf{x}_0 = \mathbf{s}, \tag{7b}$$

$$\mathbf{h}_{\boldsymbol{\theta}}\left(\mathbf{x}_k,\mathbf{u}_k\right) \le 0,\quad \mathbf{h}_{\boldsymbol{\theta}}^{\mathrm{f}}\left(\mathbf{x}_N\right) \le 0, \tag{7c}$$

where $\mathbf{u}_{0,\ldots,N-1}$, $\mathbf{x}_{0,\ldots,N}$ are the MPC input profile and predicted state trajectory. Note that, as discussed in [7], all functions defining the MPC scheme and, therefore, the policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}$, can in principle be parametrized by $\boldsymbol{\theta}$: $T_{\boldsymbol{\theta}}$, $\ell_{\boldsymbol{\theta}}$ are the terminal and stage costs, respectively, and $\mathbf{f}_{\boldsymbol{\theta}}$ the system model; functions $\mathbf{h}_{\boldsymbol{\theta}}$, $\mathbf{h}_{\boldsymbol{\theta}}^{\mathrm{f}}$ are the stage and terminal inequality constraints, respectively. We ought to stress here that the policy resulting from an MPC scheme is in general suboptimal, due to the inexact MPC model, the unmodelled stochastic disturbances impacting the real system, and the finite horizon. Even if the MPC model is obtained by performing a thorough system identification of the real system dynamics, there is not guarantee that (6) is optimal, and a further adjustment of the policy parameters can be beneficial. This remains true if the uncertainties are embedded in MPC via a stochastic or robust MPC formulation.

The use of MPC as a policy approximation in RL has been investigated and justified in [7], [8], [9], [17], [23], [24], [25], and offers some advantages over the more generic function approximations often used in RL. The main advantage is aguably that MPC-based policies allow one to impose explicitly and a priori constraints on the state trajectories predicted by the MPC model. Robust MPC techniques can then be used to ensure that the closed-loop trajectories of the real system respect these constraints, hence offering a natural path to safe RL. Moreover, MPC-based polices allow one to readily exploit the existing knowledge of the system (i.e., models and constraints) in the learning process. Furthermore, a vast set of theoretical tools are available to analyze the closed-loop behavior of MPC scheme, such as, e.g., stability and recursive feasibility. These tools are readily available to analyze and handle the closed-loop behavior of an MPC-based policy in the RL context.

It will be useful in the following to cast (7) as a generic parametric Nonlinear Program (NLP):

$$\mathbf{u}^\star\left(\mathbf{s},\boldsymbol{\theta}\right),\mathbf{x}^\star\left(\mathbf{s},\boldsymbol{\theta}\right) = \arg\min_{\mathbf{u},\mathbf{x}}\quad \Phi(\mathbf{x},\mathbf{u},\boldsymbol{\theta}) \tag{8a}$$

$$\text{s.t.}\quad \mathbf{F}\left(\mathbf{x},\mathbf{u},\mathbf{s},\boldsymbol{\theta}\right) = 0, \tag{8b}$$

$$\mathbf{H}\left(\mathbf{x},\mathbf{u},\boldsymbol{\theta}\right) \le 0. \tag{8c}$$

The theory developed in this paper will apply to (8) and will not be limited to MPC-based policies, but cover all stochastic policies generated from a smooth and well-posed NLP. Note that efficient algorithms are available to solve (8) and in this paper we are rather interested in compensating for model inaccuracies so as to recover optimality.

The use of MPC as a policy approximation has the benefits detailed above, however, it can arguably also present some challenges because the relationship between the MPC parameters $\boldsymbol{\theta}$ and the closed-loop performance $J\left(\boldsymbol{\pi}_{\boldsymbol{\theta}}\right)$ can be highly nonconvex, such that local minima can impede the learning process when following the policy gradient $\nabla_{\boldsymbol{\theta}} J\left(\boldsymbol{\pi}_{\boldsymbol{\theta}}\right)$. It shall be stressed, though, that Deep Neural Networks (DNNs), which are very popular in RL due to their effectiveness in high-dimensional spaces, also suffer from the non-convexity of $J\left(\boldsymbol{\pi}_{\boldsymbol{\theta}}\right)$ in the network parameters $\boldsymbol{\theta}$. We ought to stress that for an MPC-based policy the learning process can typically be started from a good initial guess, in the form of a reasonable system model, cost and constraints.

In contrast, good initial guesses for the weights in DNNs can be difficult to form and an initial supervised training of the DNN against a given policy can be required. In the light of these remarks, it is debatable whether the non-convexity issue is more severe for an MPC-based policy approximation than for more conventional function approximators.

A. MPC-Based Stochastic Policy

RL requires that the inputs applied to the real system undergo some exploration—i.e., that they do not consistently follow a deterministic policy $\boldsymbol{\pi_\theta}$—in order to discover directions in the policy parameters $\boldsymbol{\theta}$ that can improve closed-loop performance. Stochastic policies naturally generate this exploration. In continuous input spaces, stochastic policies are often generated via random perturbations added to a deterministic policy $\boldsymbol{\pi_\theta}$. In the context of MPC-based policies where constraints are meant to be respected, one is likely to desire that the exploration remains feasible for the MPC scheme underlying the deterministic policy $\boldsymbol{\pi_\theta}$. E.g., in the context of robust MPC, imposing that the exploration is feasible for the robust MPC scheme is necessary and sufficient (under some assumptions) to ensure that the real system trajectories respect the constraints imposed to the system, see [23]. As a result, in the MPC-based policy context, one should arguably build the stochastic policy with care, so that the feasibility of the exploration is guaranteed.

We propose to address this issue with an approach that does not add computational expenses to solving the regular MPC scheme. We propose a stochastic policy:

$$\mathbf{a} \sim \pi_{\boldsymbol{\theta}}\left[\mathbf{a} \,|\, \mathbf{s}\right] \tag{9}$$

built from $\mathbf{a} = \mathbf{u}_0^{\mathrm{d}}\left(\mathbf{s}, \boldsymbol{\theta}, \mathbf{d}\right)$ where $\mathbf{u}_0^{\mathrm{d}}$ is generated by the randomly perturbed NLP

$$\begin{aligned} \mathbf{u}^{\mathrm{d}}\left(\mathbf{s}, \boldsymbol{\theta}, \mathbf{d}\right) = \arg\min_{\mathbf{u}} \quad & \Phi^{\mathbf{d}}(\mathbf{x}, \mathbf{u}, \boldsymbol{\theta}, \mathbf{d}) && \text{(10a)} \\ \text{s.t.} \quad & \mathbf{F}\left(\mathbf{x}, \mathbf{u}, \mathbf{s}, \boldsymbol{\theta}\right) = 0, && \text{(10b)} \\ & \mathbf{H}\left(\mathbf{x}, \mathbf{u}, \boldsymbol{\theta}\right) \leq 0, && \text{(10c)} \end{aligned}$$

where the parameter $\mathbf{d} \in \mathbb{R}^m$ is drawn from an arbitrary probability density $\varrho\left(\cdot\right)$, e.g., a simple Gaussian distribution. In the following we will make the simple choice:

$$\Phi^{\mathrm{d}}(\mathbf{u}, \mathbf{s}, \boldsymbol{\theta}, \mathbf{d}) = \Phi(\mathbf{u}, \mathbf{s}, \boldsymbol{\theta}) + \mathbf{d}^\top \mathbf{u}_0. \tag{11}$$

for the perturbed cost function (10a) though alternatives can and should be considered.

By construction, a stochastic policy built from the perturbed NLP (10) is guaranteed to have all its support in the set of feasible inputs for (6). More specifically, the solutions stemming from (10) are feasible for (6) by construction, because they result from a perturbation of the cost function alone. It follows that the samples $\mathbf{a}$ drawn from (9)-(10) are guaranteed to be feasible for the MPC scheme, if the latter is recursively feasible.

Note that sampling (9) (i.e., generating a sample from density $\pi_{\boldsymbol{\theta}}\left[\mathbf{a} \,|\, \mathbf{s}\right]$) is straightforward as it requires one to only generate a sample from the chosen density $\varrho(\cdot)$ and solve the perturbed NLP (10). It is arguably a computationally cheap approach to build a stochastic policy with guaranteed feasibility. We will discuss next how to compute the score function $\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}[\mathbf{a} \,|\, \mathbf{s}]$ (required in (4)) for (9).

IV. Score function

We detail next how the score function $\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}[\mathbf{a} \,|\, \mathbf{s}]$ associated to (9)-(10) required in the policy gradient computation (4) can be computed efficiently. While sampling the stochastic policy $\pi_{\boldsymbol{\theta}}\left[\mathbf{a} \,|\, \mathbf{s}\right]$ resulting from (9)-(10) is straightforward, evaluating it (i.e., computing the value $\pi_{\boldsymbol{\theta}}\left[\mathbf{a} \,|\, \mathbf{s}\right]$ for $\mathbf{s}, \mathbf{a}, \boldsymbol{\theta}$ given) is not. For $\mathbf{s}, \mathbf{d}, \boldsymbol{\theta}$ and the sample $\mathbf{a}$ resulting from (9)-(10) given, we will show next how the score function can be computed at limited expenses.

A first issue to overcome is that, when $\mathbf{u}_0^\star$ lies at (or close to) the constraint $\mathbf{h}_{\boldsymbol{\theta}}\left(\mathbf{s}, \mathbf{u}_0\right) \leq 0$, the density $\pi_{\boldsymbol{\theta}}\left[\mathbf{a} \,|\, \mathbf{s}\right]$ from (9) can become Dirac-like on the boundary of the feasible set of $\mathbf{h}_{\boldsymbol{\theta}}\left(\mathbf{s}, \mathbf{u}_0\right) \leq 0$. More generally, this can happen whenever input $\mathbf{u}_0$ can be blocked by a constraint in the MPC scheme. This feature results from a lack of local invertibility of the mapping $\mathbf{d}$ to $\mathbf{u}_0^{\mathrm{d}}$ generated by the NLP (10) in these specific cases[1], and creates difficulties for computing the score function $\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}[\mathbf{a} \,|\, \mathbf{s}]$.

In order to alleviate this difficulty, we will cast (10) in an interior-point context. For computational reasons, we will consider the primal-dual interior point formulation of (10) [5], which have the First-Order Necessary Conditions (FONC):

$$\mathbf{r}_\tau(\mathbf{z}, \boldsymbol{\theta}, \mathbf{d}) = \begin{bmatrix} \nabla_{\mathbf{w}} \Phi^{\mathrm{d}} + \nabla_{\mathbf{w}} \mathbf{H} \boldsymbol{\mu} + \nabla_{\mathbf{w}} \mathbf{F} \boldsymbol{\lambda} \\ \mathbf{F} \\ \mathrm{diag}(\boldsymbol{\mu}) \mathbf{H} + \tau \end{bmatrix} = 0, \tag{12}$$

for $\tau > 0$ and under the conditions $\mathbf{H} < 0, \quad \boldsymbol{\mu} > 0$. Here we label $\mathbf{w} = \{\mathbf{u}, \mathbf{x}\}$ and $\mathbf{z} = \{\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}\}$ the primal-dual variables of (10). We will label $\mathbf{u}^\tau\left(\mathbf{s}, \boldsymbol{\theta}, \mathbf{d}\right)$, $\mathbf{x}^\tau\left(\mathbf{s}, \boldsymbol{\theta}, \mathbf{d}\right)$ the parametric primal solution of (12), and $\pi_{\boldsymbol{\theta}}^\tau\left[\mathbf{a}|\mathbf{s}\right]$ the stochastic policy resulting from using $\mathbf{a} = \mathbf{u}_0^\tau\left(\mathbf{s}, \boldsymbol{\theta}, \mathbf{d}\right)$ and $\mathbf{d} \sim \varrho(\cdot)$. Under standard regularity assumptions on (10), the algebraic conditions (12) admit a primal-dual solution that matches the solution of (10) with an accuracy of the order of the barrier parameter τ [5]. Moreover, the solution $\mathbf{u}^\tau\left(\mathbf{s}, \boldsymbol{\theta}, \mathbf{d}\right)$ is guaranteed to be feasible for (10). Additionally, if $\mathbf{H}, \mathbf{F}, \Phi$ are smooth, then the mapping $\mathbf{d}$ to $\mathbf{u}_0^{\mathrm{d}}$ becomes locally invertible under mild regularity conditions (LICQ and SOSC[2] [19]), as we will prove in Lemma 1.

To derive the next results, we will invert $\mathbf{u}_0^\tau\left(\mathbf{s}, \boldsymbol{\theta}, \mathbf{d}\right)$ with respect to $\mathbf{d}$, to obtain function $\boldsymbol{\zeta}(\mathbf{a}, \boldsymbol{\theta}, \mathbf{s})$ satisfying:

$$\mathbf{d} = \boldsymbol{\zeta}\left(\mathbf{u}_0^\tau\left(\mathbf{s}, \boldsymbol{\theta}, \mathbf{d}\right), \boldsymbol{\theta}, \mathbf{s}\right), \quad \forall\, \mathbf{d}. \tag{13}$$

The local existence of $\boldsymbol{\zeta}$ is guaranteed by the Implicit Function Theorem (IFT) if $\frac{\partial \mathbf{u}_0^\tau}{\partial \mathbf{d}}$ is full rank. The stochastic policy $\pi_{\boldsymbol{\theta}}$ then results from the transformation of the user-defined probability density $\varrho(\cdot)$ via NLP (10), and can be

[1] Suppose that Φ^{d} is given by (11), a constraint is $\mathbf{u} \geq 0$ and for a given $\mathbf{d}_0$ one obtains $\mathbf{u}_0^\star = 0$. Then, for $\mathbf{d}_1 > \mathbf{d}_0$ the solution is still $\mathbf{u}_0^\star = 0$.

[2] Linear Independence Constraint Qualification and strong Second-Order Sufficient Conditions.

evaluated using [4]:

$$\pi_{\boldsymbol{\theta}}\left[\mathbf{a}\,|\,\mathbf{s}\right]=\varrho\left(\boldsymbol{\zeta}\right)\left|\det\left(\frac{\partial\boldsymbol{\zeta}}{\partial\mathbf{a}}\right)\right|_{\mathbf{a},\boldsymbol{\theta},\mathbf{s}}. \tag{14}$$

For the sake of completeness, we provide hereafter a Lemma establishing that $\frac{\partial\mathbf{u}_0^\tau}{\partial\mathbf{d}}$ is full rank for the gradient perturbation strategy (11).

Lemma 1: For the choice of cost function (11), and if (10) satisfies LICQ and SOSC, the Jacobian $\frac{\partial\mathbf{u}_0^\tau}{\partial\mathbf{d}}$ of function $\mathbf{u}_0^\tau$ implicitly defined by (12) is full rank for any $\tau>0$.

Proof: For the sake of simplicity, we will prove the result using the primal interior-point conditions corresponding to (12). The lemma will then hold due to the equivalence between the primal-dual and primal interior-point solutions [19]. The primal interior-point conditions read as:

$$\left[\begin{array}{c}\nabla_{\mathbf{w}}\Phi^{\mathrm{d}}+\tau\nabla_{\mathbf{w}}\mathbf{h}\,\mathrm{diag}(\mathbf{h})^{-1}+\nabla_{\mathbf{w}}\mathbf{f}\boldsymbol{\lambda}\\ \mathbf{f}\end{array}\right]=0. \tag{15}$$

The IFT then guarantees that:

$$\left[\begin{array}{cc}H & \nabla_{\mathbf{w}}\mathbf{f}\\ \nabla_{\mathbf{w}}\mathbf{f}^\top & 0\end{array}\right]\left[\begin{array}{c}\frac{\partial\mathbf{w}}{\partial\mathbf{d}}\\ \frac{\partial\boldsymbol{\lambda}}{\partial\mathbf{d}}\end{array}\right]=-\left[\begin{array}{c}\nabla^2_{\mathbf{wd}}\Phi^{\mathrm{d}}\\ 0\end{array}\right], \tag{16}$$

where H is the Jacobian of the first row in (15) with respect to $\mathbf{w}$. Defining $\mathcal{N}$ the null space of $\nabla_{\mathbf{w}}\mathbf{f}^\top$, i.e., $\nabla_{\mathbf{w}}\mathbf{f}^\top\mathcal{N}=0$, one can verify that using $\nabla^2_{\mathbf{u}_0\mathbf{d}}\Phi^{\mathrm{d}}=I_{m\times m}$ from (11):

$$\frac{\partial\mathbf{w}}{\partial\mathbf{d}}=-\mathcal{N}\left(\mathcal{N}^\top H\mathcal{N}\right)^{-1}\mathcal{N}^\top\nabla^2_{\mathbf{wd}}\Phi^{\mathrm{d}} \tag{17}$$

$$=-\mathcal{N}\left(\mathcal{N}^\top H\mathcal{N}\right)^{-1}\mathcal{N}_0^\top, \tag{18}$$

where $\mathcal{N}_0=\left[\begin{array}{cccc}I_{m\times m} & 0 & \dots & 0\end{array}\right]\mathcal{N}$. Invertibility of $\mathcal{N}^\top H\mathcal{N}$ is follows from LICQ and SOSC of (10). Then,

$$\frac{\partial\mathbf{u}_0^\tau}{\partial\mathbf{d}}=-\mathcal{N}_0\left(\mathcal{N}^\top H\mathcal{N}\right)^{-1}\mathcal{N}_0^\top. \tag{19}$$

Since the dynamics $\mathbf{f}$ cannot restrict the input $\mathbf{u}$ in (10), $\mathcal{N}$ spans the full space of $\mathbf{u}$, and, therefore, the full input space for $\mathbf{u}_0$, such that $\mathcal{N}_0$ and (19) are full rank. ∎

One can further verify that the determinant in (14) is always positive. We can now turn to detailing how the score function $\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}[\mathbf{a}\,|\,\mathbf{s}]$ can be computed from (14). The following Lemma provides the sensitivity of $\boldsymbol{\zeta}$ required in (14).

Lemma 2: If (10) satisfies SOSC and LICQ then the following equalities hold:

$$\frac{\partial\boldsymbol{\zeta}}{\partial\boldsymbol{\theta}}=-\left(\frac{\partial\mathbf{u}_0^\tau}{\partial\mathbf{d}}\right)^{-1}\frac{\partial\mathbf{u}_0^\tau}{\partial\boldsymbol{\theta}},\quad \frac{\partial\boldsymbol{\zeta}}{\partial\mathbf{a}}=\left(\frac{\partial\mathbf{u}_0^\tau}{\partial\mathbf{d}}\right)^{-1}, \tag{20}$$

for any $\boldsymbol{\theta},\mathbf{s}$, $\mathbf{a}=\mathbf{u}_0^\tau\left(\mathbf{s},\boldsymbol{\theta},\mathbf{d}\right)$ and $\mathbf{d}=\boldsymbol{\zeta}\left(\mathbf{a},\boldsymbol{\theta},\mathbf{s}\right)$.

Proof: We observe that

$$\mathbf{u}_0^\tau\left(\mathbf{s},\boldsymbol{\theta},\boldsymbol{\zeta}\left(\mathbf{a},\boldsymbol{\theta},\mathbf{s}\right)\right)=\mathbf{a},\quad\forall\,\mathbf{a},\boldsymbol{\theta},\mathbf{s}. \tag{21}$$

The invertibility of $\frac{\partial\mathbf{g}}{\partial\mathbf{d}}$ follows from Lemma 1 and entails that the IFT applies. From the IFT, it follows that

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}}\mathbf{u}_0^\tau\left(\mathbf{s},\boldsymbol{\theta},\boldsymbol{\zeta}\left(\mathbf{a},\boldsymbol{\theta},\mathbf{s}\right)\right)=\frac{\partial\mathbf{u}_0^\tau}{\partial\boldsymbol{\theta}}+\frac{\partial\mathbf{u}_0^\tau}{\partial\mathbf{d}}\frac{\partial\boldsymbol{\zeta}}{\partial\boldsymbol{\theta}}=0, \tag{22}$$

Moreover, we observe that

$$\frac{\mathrm{d}}{\mathrm{d}\mathbf{a}}\mathbf{u}_0^\tau\left(\mathbf{s},\boldsymbol{\theta},\boldsymbol{\zeta}\left(\mathbf{a},\boldsymbol{\theta},\mathbf{s}\right)\right)=\frac{\partial\mathbf{u}_0^\tau}{\partial\mathbf{d}}\frac{\partial\boldsymbol{\zeta}}{\partial\mathbf{a}}=I, \tag{23}$$

which establishes (20). ∎

We can then use (14) to develop expressions for computing the gradient of the policy score function $\nabla_{\boldsymbol{\theta}}\log\boldsymbol{\pi}_{\boldsymbol{\theta}}$. This is detailed in the following Proposition. Computational aspects are further discussed in Section V.

Proposition 1: The gradient of the score function for $\mathbf{a}$ obtained via solving (10) from a realization of $\mathbf{d}$ reads as:

$$\nabla_\theta\log\pi_{\boldsymbol{\theta}}\left[\mathbf{a}\,|\,\mathbf{s}\right]=\mathbf{m}-\left(\varrho^{-1}\frac{\partial\varrho}{\partial\mathbf{d}}\left(\frac{\partial\mathbf{u}_0^\tau}{\partial\mathbf{d}}\right)^{-1}\frac{\partial\mathbf{u}_0^\tau}{\partial\boldsymbol{\theta}}\right)^\top, \tag{24}$$

evaluated at $\mathbf{s},\boldsymbol{\theta},\mathbf{d}$, where the components of vector $\mathbf{m}$ are:

$$\mathbf{m}_i=\mathrm{Tr}\left(\frac{\partial\mathbf{u}_0^\tau}{\partial\mathbf{d}}\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}_i}\frac{\partial\boldsymbol{\zeta}}{\partial\mathbf{a}}\right)\Bigg|_{\mathbf{s},\boldsymbol{\theta},\mathbf{d},\mathbf{a}}. \tag{25}$$

Proof: Using (14), the score function of $\pi_{\boldsymbol{\theta}}\left[\mathbf{a}\,|\,\mathbf{s}\right]$ is:

$$\log\pi_{\boldsymbol{\theta}}\left[\mathbf{a}\,|\,\mathbf{s}\right]=\log\varrho\left(\boldsymbol{\zeta}\right)-\log\det\left(\frac{\partial\boldsymbol{\zeta}}{\partial\mathbf{a}}\right). \tag{26}$$

Using (20) we observe that:

$$\nabla_{\boldsymbol{\theta}}\log\varrho\left(\boldsymbol{\zeta}\right)=\left(\varrho^{-1}\frac{\partial\varrho}{\partial\mathbf{d}}\frac{\partial\boldsymbol{\zeta}}{\partial\boldsymbol{\theta}}\right)^\top\Bigg|_{\mathbf{s},\boldsymbol{\theta},\mathbf{d},\mathbf{a}} \tag{27}$$

$$=-\left(\varrho^{-1}\frac{\partial\varrho}{\partial\mathbf{d}}\left(\frac{\partial\mathbf{u}_0^\tau}{\partial\mathbf{d}}\right)^{-1}\frac{\partial\mathbf{u}_0^\tau}{\partial\boldsymbol{\theta}}\right)^\top\Bigg|_{\mathbf{s},\boldsymbol{\theta},\mathbf{d}},$$

hence providing the second term in (24). From calculus and using (20), we get:

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}_i}\log\det\left(\frac{\partial\boldsymbol{\zeta}}{\partial\mathbf{a}}\right)=\mathrm{Tr}\left(\left(\frac{\partial\boldsymbol{\zeta}}{\partial\mathbf{a}}\right)^{-1}\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}_i}\frac{\partial\boldsymbol{\zeta}}{\partial\mathbf{a}}\right) \tag{28}$$

$$=\mathrm{Tr}\left(\frac{\partial\mathbf{u}_0^\tau}{\partial\mathbf{d}}\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}_i}\frac{\partial\boldsymbol{\zeta}}{\partial\mathbf{a}}\right)=\mathbf{m}_i,$$

hence providing (25) component-wise. ∎

V. Computational aspects

We now turn to detailing how the sensitivities of functions $\mathbf{u}_0^\tau$ and $\boldsymbol{\zeta}$ required in (24)-(25) can be computed at a limited computational cost. First, it is useful to provide the sensitivities of function $\boldsymbol{\zeta}$. If LICQ and SOSC hold [19] for NLP (10), one can verify that the IFT guarantees that for $\mathbf{r}_\tau$ readily obtained from (12):

$$\frac{\partial\mathbf{r}_\tau}{\partial\mathbf{z}}\frac{\partial\mathbf{z}}{\partial\mathbf{d}}+\frac{\partial\mathbf{r}_\tau}{\partial\mathbf{d}}=0,\qquad\frac{\partial\mathbf{r}_\tau}{\partial\mathbf{z}}\frac{\partial\mathbf{z}}{\partial\boldsymbol{\theta}}+\frac{\partial\mathbf{r}_\tau}{\partial\boldsymbol{\theta}}=0, \tag{29}$$

and therefore $\frac{\partial\mathbf{u}_0^\tau}{\partial\mathbf{d}}$ and $\frac{\partial\mathbf{u}_0^\tau}{\partial\boldsymbol{\theta}}$, required in the second term of (24), can be extracted as the first components of $\frac{\partial\mathbf{z}}{\partial\mathbf{d}}$ and $\frac{\partial\mathbf{z}}{\partial\boldsymbol{\theta}}$ obtained by solving the linear systems (29).

Obtaining the second-order term $\frac{\partial^2\boldsymbol{\zeta}}{\partial\boldsymbol{\theta}_i\partial\mathbf{a}}$ in (28) can be fairly involved. In order to simplify its computation, we propose to use the following approach. Let us define:

$$\tilde{\mathbf{z}}=\left\{\mathbf{d},\,\mathbf{u}_1,\,\dots,\,\mathbf{u}_{N-1},\mathbf{x},\,\boldsymbol{\lambda},\boldsymbol{\mu}\right\}, \tag{30}$$

One can observe that $\tilde{\mathbf{z}}$ results from replacing variable $\mathbf{u}_0$ with $\mathbf{d}$ in $\mathbf{z}$. One can then verify that the sensitivities of $\boldsymbol{\zeta}$ can be obtained from considering $\mathbf{u}_0$ as a parameter in (12)

and $\mathbf{d}$ as part of the solution, becoming an implicit function of $\mathbf{u}_0$. The IFT then naturally applies and delivers the first-order sensitivities:

$$\frac{\partial \mathbf{r}_\tau}{\partial \tilde{\mathbf{z}}}\frac{\partial \tilde{\mathbf{z}}}{\partial \mathbf{a}} + \frac{\partial \mathbf{r}_\tau}{\partial \mathbf{u}_0} = 0, \qquad \frac{\partial \mathbf{r}_\tau}{\partial \tilde{\mathbf{z}}}\frac{\partial \tilde{\mathbf{z}}}{\partial \boldsymbol{\theta}} + \frac{\partial \mathbf{r}_\tau}{\partial \boldsymbol{\theta}} = 0, \tag{31}$$

and second-order sensitivities:

$$\frac{\partial \mathbf{r}_\tau}{\partial \tilde{\mathbf{z}}}\frac{\partial^2 \tilde{\mathbf{z}}}{\partial \boldsymbol{\theta}_i \partial \mathbf{a}} + \left(\frac{\partial^2 \mathbf{r}_\tau}{\partial \boldsymbol{\theta}_i \partial \tilde{\mathbf{z}}} + \sum_j \frac{\partial^2 \mathbf{r}_\tau}{\partial \tilde{\mathbf{z}} \partial \mathbf{z}_j}\frac{\partial \tilde{\mathbf{z}}_j}{\partial \boldsymbol{\theta}_i} \right)\frac{\partial \tilde{\mathbf{z}}}{\partial \mathbf{a}} + \frac{\partial^2 \mathbf{r}_\tau}{\partial \boldsymbol{\theta}_i \partial \mathbf{a}} + \sum_j \frac{\partial^2 \mathbf{r}_\tau}{\partial \mathbf{a} \partial \tilde{\mathbf{z}}_j}\frac{\partial \tilde{\mathbf{z}}_j}{\partial \boldsymbol{\theta}_i} = 0. \tag{32}$$

The second-order term $\frac{\partial^2 \boldsymbol{\zeta}}{\partial \boldsymbol{\theta}_i \partial \mathbf{a}}$ in (28) can be extracted from solving the linear system (32) for $\frac{\partial^2 \tilde{\mathbf{z}}}{\partial \boldsymbol{\theta}_i \partial \mathbf{a}}$ and observing that $\frac{\partial^2 \boldsymbol{\zeta}}{\partial \boldsymbol{\theta}_i \partial \mathbf{a}} = \frac{\partial^2 \mathbf{d}}{\partial \boldsymbol{\theta}_i \partial \mathbf{a}}$ is the first element of $\frac{\partial^2 \tilde{\mathbf{z}}}{\partial \boldsymbol{\theta}_i \partial \mathbf{a}}$.

In summary, computing the score function, requires then one to form and solve the linear systems (29)-(32), and to use the resulting sensitivities in (24)-(25). We ought to underline here that the linear systems (29)-(32) are typically large but very sparse if coming from an MPC scheme. If their sparsity is correctly exploited when forming and solving the linear systems, the computational burden remains fairly small, typically a fraction of the computational burden associated with the solution of the MPC poolicy.

VI. Illustrative Example

This section presents a brief academic example where the proposed method is applied. The example is simple so as to present results that can be easily interpreted, and we do not claim that the problem could not be treated by alternative methods. We consider the system:

$$\mathbf{s}_+ = \begin{bmatrix} 1 & \frac{1}{10} \\ 0 & 1 \end{bmatrix} \mathbf{s} + \begin{bmatrix} \frac{1}{10} & 0 \\ 0 & 1 \end{bmatrix} \mathbf{a} + \mathbf{e}, \tag{33}$$

where $\mathbf{e} \sim \mathcal{N}\left(0, 10^{-2} I\right)$ is a stochastic perturbation that we assume unknown. The baseline cost was chosen as:

$$L(\mathbf{s}, \mathbf{a}) = \|\mathbf{s} - \bar{\mathbf{s}}\|^2 + \|\mathbf{a} - \bar{\mathbf{a}}\|^2 + 10 \cdot \mathbf{1}^\top \max\left(0, \mathbf{h}\left(\mathbf{s}\right)\right), \tag{34}$$

hence imposing a linear penalty on the state constraints $\mathbf{h}\left(\mathbf{s}\right) \leq 0$ being violated, where

$$\mathbf{h}\left(\mathbf{s}\right)^\top = \begin{bmatrix} -\mathbf{s}_1 & -\mathbf{s}_2 & \mathbf{s}_1 - 1 & \mathbf{s}_2 - 1 \end{bmatrix} \tag{35}$$

imposes bounds on the states. The MPC scheme reads as:

$$\min_{\mathbf{x}, \mathbf{u}, \boldsymbol{\sigma}, \boldsymbol{\delta}} \ \|\mathbf{x}_N - \bar{\mathbf{x}}\|_Q^2 + \sum_{k=0}^{N-1} \ell_{\boldsymbol{\theta}}\left(\mathbf{x}_k, \mathbf{u}_k\right) + 10 \cdot \mathbf{1}^\top \boldsymbol{\sigma}_k \tag{36a}$$

$$\text{s.t. } \mathbf{x}_{k+1} = \begin{bmatrix} 1 & \frac{1}{10} \\ 0 & 1 \end{bmatrix} \mathbf{x}_k + \begin{bmatrix} \frac{1}{10} & 0 \\ 0 & 1 \end{bmatrix} \mathbf{u}_k \tag{36b}$$

$$\mathbf{h}\left(\mathbf{x}_k\right) + \boldsymbol{\delta}_k \leq \boldsymbol{\sigma}_k, \quad \boldsymbol{\sigma}_k \geq 0 \tag{36c}$$

$$\boldsymbol{\delta}_{k+1} = \operatorname{diag}(\mathbf{c})\boldsymbol{\delta}_k + \mathbf{b} \tag{36d}$$

$$\mathbf{x}_0 = \mathbf{s}, \quad \boldsymbol{\delta}_0 = 0 \tag{36e}$$

with $N = 30$ and where $\ell_{\boldsymbol{\theta}} = \|\mathbf{x} - \tilde{\mathbf{x}}\|^2 + \|\mathbf{u} - \tilde{\mathbf{u}}\|^2$ and $\boldsymbol{\theta} = \{\mathbf{c}, \mathbf{b}, \tilde{\mathbf{x}}, \tilde{\mathbf{u}}\}$ is the set of parameters adjustable by RL. One can readily observe that for $\mathbf{c}, \mathbf{b} > 0$, (36c)-(36d) performs a form of tightening of the state constraints, implicitly accounting for the stochasticity of the system. If the support of $\mathbf{e}$ is bounded and can be enclosed in a known polytope, then efficient techniques exist to compute a tightening ensuring that no violation of the constraints can happen [12]. Here, the constraint relaxation penalties chosen as 10 is not very large, such that occasional violations of $\mathbf{h}\left(\mathbf{s}\right) \leq 0$ are beneficial in regard of the baseline cost (34). RL then uses data to discover the right amount of constraint tightening and possibly reference changes to achieve the optimal amount of constraints violation, without an actual knowledge of the process noise $\mathbf{e}$. Guaranteed constraint satisfaction is not the objective in this formulation, we refer to the robust MPC techniques in [23], which can be readily combined with the approach presented here to tackle problems with constraints that should never be violated.

A rudimentary RL strategy was used, where the value function $V_{\pi_{\boldsymbol{\theta}}}$ is approximated via a quadratic function, and estimated via Least-Squares TD on batches of 50 time samples. The policy gradient is computed on the same batches, without experience replay. A discount of $\gamma = 0.9$ was chosen. No step size adaptation was used so a fairly small step size of 10^{-3} was selected. The density ϱ was chosen as Gaussian of standard deviation 10^{-3}, and the barrier parameter was chosen as $\tau = 10^{-2}$. A non-episodic scenario was considered. The infeasible baseline reference $\bar{\mathbf{s}}^\top = \begin{bmatrix} -10^{-1} & 0 \end{bmatrix}$ was used with $\bar{\mathbf{a}} = 0$. The RL steps where projected on the constraint $\mathbf{b} \geq 0$, such that positive backoffs are ensured. A much higher sample efficiency could clearly be achieved via more advanced algorithmic setups, but the focus here was on simplicity.

Fig. 1 depicts the (relative) evolution of the cost associated to each batch over the learning process. Fig. 2 depicts the evolution of the MPC parameters $\mathbf{c}, \mathbf{b}$ handling the constraint tightening. Fig. 3 and 4 depict the evolution of states $\mathbf{s}_1, \mathbf{s}_2$ and inputs $\mathbf{a}_1, \mathbf{a}_2$, together with the backoff parameters $\mathbf{b}$ in Fig. 3 and the evolution of the MPC references $\tilde{\mathbf{x}}$, $\tilde{\mathbf{u}}$ in both Fig. 3 and 4. All horizontal axis have the unit of time samples, RL steps are taken every 50 time samples. One can observe the reduction of the cost J, albeit it is noisy due to the process noise. The increase of closed-loop performance is achieved via removing the constraint tightening while adjusting MPC references in a non-trivial combination. The tightening time constants $\mathbf{c}$ are not changed much. Due to the high noise in the cost function J, a direct policy search is likely to be difficult for this example.

VII. Conclusion

This paper investigates the use of the Actor-Critic, stochastic policy gradient method on MPC-based policies. If it is desired that the inputs applied to the system are always feasible with respect to the MPC scheme, then the stochastic policy must be constructed carefully. This paper proposes a very simple and inexpensive technique to do so. The paper further details how the score function associated to the MPC-based policy can be evaluated in view of computing

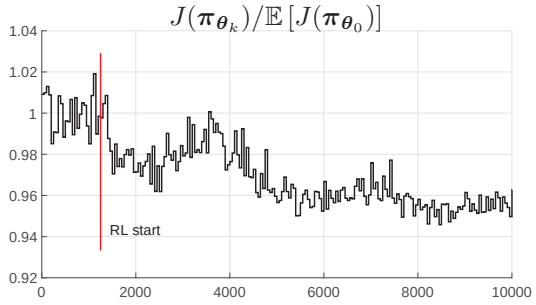

Fig. 1. Progression of the (normalized) cost over the RL steps, evaluated on the batches of 50 samples (without discount). The learning starts at the vertical red line.

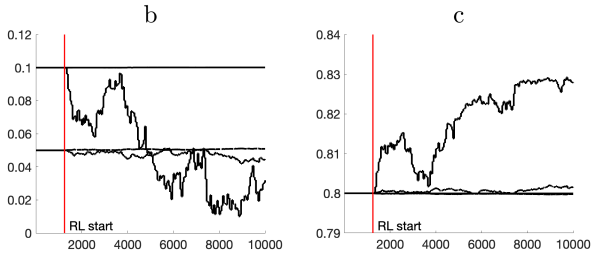

Fig. 2. MPC parameters $\boldsymbol{\theta}$ over the RL steps. The left graph depicts $\mathbf{b}$ and the right graph depicts $\mathbf{c}$. The learning starts at the vertical red line.

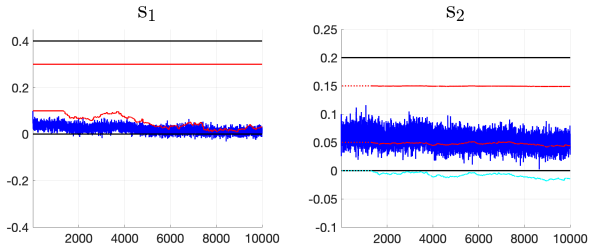

Fig. 3. States $\mathbf{s}_1, \mathbf{s}_2$ in blue over the learning process, bounds in black and the constraint backoff $\mathbf{b}$ in red. The light-blue lines represent the references $\tilde{\mathbf{x}}$ ($\tilde{\mathbf{x}}_1$ is not moving).

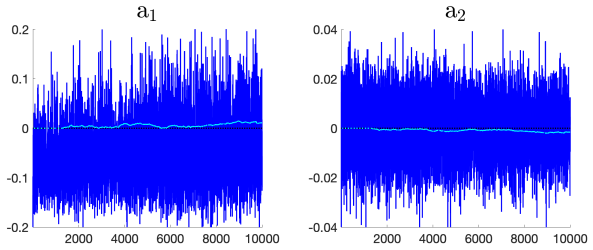

Fig. 4. Inputs $\mathbf{a}$ over the learning process. The light-blue lines represent the references $\tilde{\mathbf{u}}$

the policy gradient. Principles from parametric Nonlinear Programming and the primal-dual interior point methods are required to achieve that. The computational aspects required to implement the proposed approach are provided. The paper concludes with a simple example, where the constraints are tightened in the MPC scheme and adjusted via RL in order to achieve the optimal amount of violations in regard to the constraints relaxation in use. Techniques combining robust MPC and RL [23], can arguably be readily combined to the policy gradient approach presented here to tackle problems where constraints should never be violated.

REFERENCES

[1] P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng. An application of reinforcement learning to aerobatic helicopter flight. In *NIPS 19*, page 2007. MIT Press, 2007.

[2] B. Amos, I. D. J. Rodriguez, J. Sacks, B. Boots, and J. Z. Kolter. Differentiable mpc for end-to-end planning and control. In *NIPS*, pages 8299–8310, USA, 2018.

[3] D.P. Bertsekas. *Reinforcement Learning and Optimal Control*. Athena Scientific, 2009.

[4] D.P. Bertsekas and J.N. Tsitsiklis. *Optimum Experimental Designs*. Mass. : Athena Scientific. Belmont, 2002.

[5] L. T. Biegler. *Nonlinear Programming*. MOS-SIAM Series on Optimization. SIAM, 2010.

[6] S. Chen, K. Saulnier, N. Atanasov, D. D. Lee, V. Kumar, G. J. Pappas, and M. Morari. Approximating explicit model predictive control using constrained neural networks. In *2018 Annual American Control Conference (ACC)*, pages 1520–1527, June 2018.

[7] S. Gros and M. Zanon. Data-Driven Economic NMPC Using Reinforcement Learning. *IEEE Transactions on Automatic Control*, 65(2):636–648, Feb 2020.

[8] S. Gros and M. Zanon. Reinforcement Learning for Mixed-Integer Problems Based on MPC. In *21st IFAC World Congress*, 2020.

[9] S. Gros, M. Zanon, and A. Bemporad. Safe Reinforcement Learning via Projection on a Safe Set: How to Achieve Optimality? In *21st IFAC World Congress*, 2020.

[10] M. Hertneck, J. K., S. Trimpe, and F. Allgöwer. Learning an approximate model predictive controller with guarantees. *IEEE Control Systems Letters*, 2:543–548, 2018.

[11] T. Koller, F. Berkenkamp, M. Turchetta, and A. Krause. Learning-based Model Predictive Control for Safe Exploration and Reinforcement Learning. Published on Arxiv, 2018.

[12] W. Langson, S. Rakovic I. Chryssochoos, and D. Mayne. Robust model predictive control using tubes. *Automatica*, 40:125–133, 2004.

[13] F. L. Lewis and D. Vrabie. Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE Circuits and Systems Magazine*, 9(3):32–50, 2009.

[14] F. L. Lewis, D. Vrabie, and K. G. Vamvoudakis. Reinforcement learning and feedback control: Using natural decision methods to design optimal adaptive controllers. *IEEE Control Systems*, 32(6):76–105, 2012.

[15] S. Lucia and B. Karg. A deep learning-based approach to robust nonlinear model predictive control. *IFAC-PapersOnLine*, 51(20):511 – 516, 2018. IFAC Conf. on NMPC.

[16] T. Mannucci, E. van Kampen, C. de Visser, and Q. Chu. Safe exploration algorithms for reinforcement learning controllers. *IEEE Trans. on Neural Networks and Learning Systems*, 29(4):1069–1081, 2018.

[17] A. B. Martinsen, A. M. Lekkas, and S. Gros. Combining Ssystem Identification with Reinforcement Learning-Based MPC. In *21st IFAC World Congress*, 2020. (accepted).

[18] R. Murray and M. Palladino. A model for system uncertainty in reinforcement learning. *Systems & Control Letters*, 122:24 – 31, 2018.

[19] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer, 2006.

[20] C. J. Ostafew, A. P. Schoellig, and T. D. Barfoot. Robust Constrained Learning-based NMPC enabling reliable mobile robot path tracking. *Int. Journal of Robotics Research*, 35(13):1547–1563, 2016.

[21] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, USA, 2nd edition, 2018.

[22] S. Wang, W. Chaovalitwongse, and R. Babuska. Machine learning algorithms in bipedal robot control. *Trans. Sys. Man Cyber Part C*, 42(5):728–743, September 2012.

[23] M. Zanon and Gros. Safe Reinforcement Learning Using Robust MPC. *Transaction on Automatic Control, (in press)*, 2021.

[24] M. Zanon, S. Gros, and A. Bemporad. Practical Reinforcement Learning of Stabilizing Economic MPC. In *Proceedings of the European Control Conference*, 2019.

[25] M. Zanon, V. Kungurtsev, and S. Gros. Reinforcement Learning Based on Real-Time Iteration NMPC. In *21st IFAC World Congress*, 2020.