2024 European Control Conference (ECC)
June 25-28, 2024. Stockholm, Sweden

Predictive Control with Terminal Costs Based on Online Learning Using Value Iteration

Francisco Moreno-Mora and Stefan Streif

Abstract— **In this work, a predictive controller that uses online-learned terminal costs is proposed. The learned costs are based on Approximate Dynamic Programming (ADP), specifically, Value Iteration (VI), an ADP technique that aims to iteratively determine the optimal value function of an optimal control problem. With this, we aim to improve infinite-horizon controller performance. Instead of performing an offline iteration over the whole state space, we consider a local update law which is executed online and reduces the computational burden. We first extend results on local VI to the case where the iteration is initialized with the value function of a stabilizing feedback policy, showing that the local update law preserves the stability of the associated control law. Then, we use the approximated cost function in a predictive controller framework and provide recursive feasibility, stability guarantees and an estimate of the region of attraction for a sufficiently long prediction horizon. The proposed approach is evaluated in simulation against a predictive controller which uses VI over the whole state space, and a predictive controller without terminal costs, to show the advantages of the proposed controller.**

I. Introduction

Model Predictive Control (MPC) is a control approach based on an optimization problem which involves the minimization at each sampling time of a cost function along a prediction of the system trajectory, starting from the current system measurement. Constraints can be included directly in the optimization problem to guarantee their satisfaction. To ensure the stability of the closed-loop, so-called terminal ingredients, i.e. terminal cost and terminal set, are commonly used [14]. The terminal cost may be derived by using a local linear quadratic regulation approach based on the linearized dynamics [3] or constructed by a finite-horizon or infinite-horizon cost function under a known control law (see [4] and [9], [13], respectively). Learning-based methods, which commonly use a parametric approximation of an optimal infinite-horizon cost function, may also be considered to design the terminal cost, e.g. [1], [15]. These results provide performance guarantees with respect to the infinite-horizon optimal control. The approximation of the value function is based on Approximate Dynamic Programming (ADP).

In ADP, function approximators are used to solve optimal control problems. For infinite-horizon optimal control problems, this provides a way of alleviating the curse of dimensionality [6]. Two important methods can be distinguished in the ADP literature, namely Value Iteration (VI) and Policy Iteration (PI) (see [10], [16] for an overview). One difference between the two is that PI requires an initial policy that is stabilizing. The policies resulting from the iterations also remain stabilizing, which is of interest for online iterations [7]. However, it has also been shown that initializing VI with the value function of a stabilizing policy, guarantees that the value functions generated by the iteration correspond to value functions of stabilizing policies [7]. Convergence analysis of VI can be found in, e.g., [11] and [2], and for PI in [12] and [2].

Implementing ADP methods typically entails solving a pointwise iteration over the whole domain of interest which can be very computationally demanding and prohibits the use of these methods for online applications. Furthermore, as the system complexity increases, so does the amount of data necessary to train the approximated functions. Considering the fact that a system typically operates in a subset of the state space, gathering data for the entire state space may not be possible. A local version of VI was proposed in [17] to address the aforementioned drawbacks. The properties of the learned value function and the convergence of the algorithm were also analyzed.

In this paper we extend the results of [17] using similar arguments as in [7] to the case where the algorithm is initalized using the value function of a stabilizing policy. We analyze the properties of the approximated value function, with particular focus on the stability of the associated feedback law. We show that local VI preserves the stability of this feedback law, like its global counterpart. Furthermore, following the ideas of [15], we embed the learned function in an MPC framework to exploit the learned value function to improve infinite-horizon closed-loop performance. Then, only samples in the vicinity of the predicted terminal cost can be used for the update of the learned function, reducing the computational burden. By using a long-enough prediction horizon we can guarantee stability and recursive feasibility and give an estimate of the region of attraction of the controller.

We first define the problem setup and introduce stabilizing VI and local VI. Then, we present our results for local stabilizing VI and the closed-loop stability analysis in Section III. In Section IV we implement the predictive controller based on local stabilizing VI for the orbital rendezvous problem and compare it with global VI and an MPC controller without terminal cost. Section V concludes the paper.

Notation. We denote the set of integers greater or equal to an integer i by $\mathbb{Z}_i$ and the set of non-negative real numbers by $\mathbb{R}_{\geq 0}$. The ceiling function is denoted by $\lceil \cdot \rceil$. $\mathcal{K}_\infty$ denotes the set of functions $\alpha : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ that are continuous, strictly increasing, unbounded and satisfy $\alpha(0) = 0$. The Kronecker

Francisco Moreno-Mora and Stefan Streif are with the Laboratory for Automatic Control and System Dynamics, Technische Universitat Chemnitz, 09107 Chemnitz, Germany.

product is denoted by $\otimes$ and $\text{diag}(x_1, \dots, x_n)$ is a diagonal matrix with main diagonal entries $x_1, \dots, x_n$. We denote the exponential function as $\exp(x)$.

II. Problem setup and preliminaries

Consider a discrete-time nonlinear system

$$x(k+1) = f(x(k), u(k)), \quad k \in \mathbb{Z}_0 \tag{1}$$

where $x(k) \in \mathbb{R}^n$ is the state and $u(k) \in \mathbb{R}^m$ is the control input, with initial value $x(0) = x_0$. The system dynamics f, satisfy $f(0,0) = 0$ and are continuous. Furthermore, the system is subject to constraints

$$(x(k), u(k)) \in \mathbb{D} \coloneqq \mathbb{X} \times \mathbb{U} \subset \mathbb{R}^n \times \mathbb{R}^m, \quad k \in \mathbb{Z}_0,$$

with a compact $\mathbb{D}$, for which $0 \in \text{int}(\mathbb{D})$. As control goal, we aim to minimize the closed-loop cost associated to a continuous stage cost of the form $l : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}_{\geq 0}$

$$l(x,u) = Q(x) + u^\top R u, \quad Q : \mathbb{R}^n \to \mathbb{R}_{\geq 0}, \quad R \succ 0.$$

We require the following assumption, which is typically fulfilled by choosing quadratic stage costs.

Assumption 1: There exist functions $\underline{\alpha}_l$, $\overline{\alpha}_l \in \mathcal{K}_\infty$, such that $\underline{\alpha}_l(\|x\|) \leq l(x,0) \leq \overline{\alpha}_l(\|x\|)$ for all $x \in \mathbb{R}^n$.

In the next section we introduce VI, upon which our predictive controller is based.

A. Value Iteration

VI is a method to approximate the solution of nonlinear optimal control problems. Consider the infinite-horizon cost (2) for some $x \in \mathbb{R}^n$ and a sequence $\overline{u}(\cdot) \in \mathbb{U}^\infty$, where $\mathbb{U}^\infty$ denotes the set of sequences that fulfill some, possibly state-dependent, input constraints $\overline{u}(k) \in \mathbb{U}(\overline{x}(k))\ \forall k \in \mathbb{Z}_0$,

$$J(x, \overline{u}(\cdot)) \coloneqq \sum_{k=0}^{\infty} l(\overline{x}(k), \overline{u}(k)), \tag{2}$$

with $\overline{x}(k+1) = f(\overline{x}(k), \overline{u}(k))$, $\overline{x}(0) = x$. Assuming the minimum of this cost exists, we define the optimal value function as

$$V(x) \coloneqq \min_{\overline{u}(\cdot) \in \mathbb{U}^\infty} J(x, \overline{u}(\cdot)).$$

The value function satisfies the Bellman equation

$$V(x) = \min_{u \in \mathbb{U}(x)} \left(l(x,u) + V(f(x,u)) \right),$$

with the associated optimal controller given by

$$h^*(x) = \arg\min_{u \in \mathbb{U}(x)} \left(l(x,u) + V(f(x,u)) \right).$$

Remark 1: This section considers only input constraints for simplicity. However, state constraints can be translated into state-dependent input constraints by using the dynamics of the system.

Definition 1: (Adapted from [6]) An admissible controller within a set $\Omega \subset \mathbb{R}^n$, containing the origin, is defined as a continuous law $u : \Omega \to \mathbb{R}^m$ with $u(0) = 0$, such that $J(x, \overline{u}(\cdot))$, with $\overline{u}(k) = u(\overline{x}(k))$, $k \in \mathbb{Z}_0$, is finite for any $x \in \Omega$ and $\overline{u}(k) \in \mathbb{U}(\overline{x}(k))$ for all $k \in \mathbb{Z}_0$.

To ensure boundedness of the value function on Ω, existence of at least one admissible controller is required. We consider a set $\Omega \subset \mathbb{R}^n$ compact with $0 \in \text{int}(\Omega)$.

Assumption 2: There exists an admissible controller for system (1) on Ω.

VI describes an iterative procedure to approximate the optimal value function over a domain of interest, in our case Ω, as introduced in Assumption 2. For this purpose, the following iteration is performed

$$V_{i+1}(x) = \min_{u \in \mathbb{U}(x)} \left(l(x,u) + V_i(f(x,u)) \right), \quad \forall x \in \Omega, \tag{3}$$

using an approximator $V_i : \Omega \to \mathbb{R}_{\geq 0}$ over $i \in \mathbb{Z}_0$, with initial guess V_0. We associate a controller with each V_i as follows

$$h_i(x) \coloneqq \arg\min_{u \in \mathbb{U}(x)} \left(l(x,u) + V_i(f(x,u)) \right), \quad \forall x \in \Omega.$$

The approximated function V_i may have desired properties if V_0 is initialized properly, as shown next.

B. Stabilizing Value Iteration

When VI as defined in (3) is initialized using the value function of an admissible control policy within a set $\Omega \subset \mathbb{R}^n$, it is often referred to as *stabilizing* VI. Obtaining the value function (V_π) of a admissible policy (π) amounts to solving the equation

$$V_\pi(x) = l(x, \pi(x)) + V_\pi\left(f(x, \pi(x))\right), \quad \forall x \in \Omega. \tag{4}$$

We recall two results from [7], which will play an important role in the stability analysis of the proposed controller.

Lemma 1: Let Assumption 2 hold. The sequence of functions $\{V_j(x)\}_{j=0}^{\infty} \coloneqq \{V_0(x), V_1(x), \dots\}$ generated through stabilizing VI is pointwise non-increasing in Ω.

Furthermore, using an admissible initial policy ensures that the policies associated to the approximated value functions $V_i(x)$ are also admissible as shown by the next result.

Theorem 1: Let Assumption 2 hold and assume that only the origin is a solution for the equations $x = f(x,0)$ and $l(x,0) = 0$. For every fixed $i \in \mathbb{Z}_1$, the control policy $h_i(\cdot)$ generated using stabilizing value iteration renders the origin an asymptotically stable point. Moreover, the set $\beta_r^i \coloneqq \{x \in \mathbb{R}^n \ : \ V_i(x) \leq r\}$ for any $r > 0$ such that $\beta_r^i \subset \Omega$, is a subset of the region of attraction of the closed-loop system.

However, the iteration described by (3) must be performed *over the entire* state space. Depending on the application this may not be possible, such as when the behavior of the system is only known for a subset of the state space or when the algorithm is implemented online [17]. To avoid this, a local value iteration scheme can be used, introduced next.

C. Local Value Iteration

The following VI scheme and the results presented in this section were proposed in [17]. Consider a sequence of sets $\{\mathcal{L}_i\}$ for $i \in \mathbb{Z}_0$, where for each set it holds that $\mathcal{L}_i \subseteq \Omega$, and define a sequence of functions $\{\lambda_i(x)\}$, where each function is such that

$$\begin{cases} 0 < \lambda_i(x) \leq 1, & \forall x \in \mathcal{L}_i, \\ \lambda_i(x) = 0, & \forall x \in \Omega \setminus \mathcal{L}_i. \end{cases}$$

These sets determine where VI is applied at each time step and need not be related to one another in some specific way. Then, for $i = 1, 2, ...$ and $\forall x \in \mathcal{L}_{i-1}$, the approximation is locally updated as

$$\Gamma_i(x) \coloneqq \min_u \ (l(x,u) + V_{i-1}(f(x,u))). \tag{5}$$

Finally, the global approximation is computed as

$$V_i(x) = (1 - \lambda_{i-1}(x))V_{i-1}(x) + \lambda_{i-1}(x)\Gamma_i(x), \ \forall x \in \Omega. \tag{6}$$

Thus, at each iteration i, the approximation is updated only for points in the set $\mathcal{L}_{i-1}$. The initial guess V_0 is selected as an arbitrary positive semidefinite function. We can associate a controller to each V_i as follows

$$\pi_i(x) = \arg\min_u \ (l(x,u) + V_i(f(x,u))). \tag{7}$$

Remark 2: The local VI scheme was originally proposed without constraints, as presented above. However, we will include henceforth input constraints in the minimization as done in Section II-A.

The following theorem establishes the convergence of the local value iteration scheme.

Theorem 2: Let V_i be calculated as in (6). If for all $x \in \Omega$, the learning rate function $\lambda_i(x)$ satisfies

$$\sum_{i=0}^{\infty} \lambda_i(x) = \infty, \ \forall x \neq 0$$

then the iterative value function V_i converges to the optimal performance index function V as $i \to \infty$.

Note that condition on λ_i requires that any point in Ω is sampled infinitely many times, which goes against the objective of adapting the approximation *only* along the predicted trajectory of the system. Thus, in the next section we show that using the value function of an admissible policy to initialize the local one, we can preserve the stability of the associated control law. Then, we can use any of the approximations generated by the algorithm as local Lyapunov functions for the system.

III. Main Results

A. Local stabilizing VI

We first extend the results of Section II-C to the case where the iteration is initialized with the value function of a admissible policy. Because of the input constraints in our problem setup, we consider henceforth that the minimization in (5) and (7) is subject to the constraints presented in Section II-A. We first show that the sequence V_i is monotonically nondecreasing in the following lemma.

Lemma 2: Let the local VI (6) be initialized with the value function of an admissible policy, then for any $i \in \mathbb{Z}_0$ and for all $x \in \Omega$, the sequence V_i is monotonically non-increasing, i.e.

$$V_{i+1}(x) \leq V_i(x). \tag{8}$$

Proof: We use similar arguments as in [17, Theorem 5]. First, we show that the statement is true for the first iteration. For $x \in \mathcal{L}_0$ it holds that

$$\begin{aligned} V_1(x) &= (1 - \lambda_0(x))V_0(x) + \lambda_0(x)\Gamma_1(x) \\ &\leq (1 - \lambda_0(x))V_0(x) + \lambda_0(x)V_0(x) = V_0(x), \end{aligned}$$

where the inequality holds because V_0 is the value function of a admissible policy (see II-B). For $x \in \Omega \setminus \mathcal{L}_0$, $V_1(x) = V_0(x)$. Thus, the statement is verified for the first iteration. Similarly, one can also show that $\Gamma_1(x) \leq V_0(x), \ \forall x \in \Omega$. Henceforth we assume that

$$\Gamma_{i+1}(x) \leq V_i(x), \ \forall x \in \Omega, i \in \{0, \dots, l-1\}, l \in \mathbb{Z}_1.$$

Then, for all x in Ω we obtain

$$\begin{aligned} V_l(x) &= (1 - \lambda_{l-1}(x))V_{l-1}(x) + \lambda_{l-1}(x)\Gamma_l(x) \\ &\geq (1 - \lambda_{l-1}(x))\Gamma_l(x) + \lambda_{l-1}(x)\Gamma_l(x) = \Gamma_l(x). \end{aligned}$$

The final statement is proven by induction. Let (8) hold for $i \in \{0, \dots, l-1\}, \ l \in \mathbb{Z}_1$. Then, for $x \in \Omega$ it holds that

$$\begin{aligned} \Gamma_{l+1}(x) &= \min_{u \in \mathbb{U}(x)} (l(x,u) + V_l(f(x,u))) \\ &\leq \min_{u \in \mathbb{U}(x)} (l(x,u) + V_{l-1}(f(x,u))) = \Gamma_l(x). \end{aligned}$$

By induction, we conclude that

$$\Gamma_{i+1}(x) \leq \Gamma_i(x), \quad \forall x \in \Omega, \tag{9}$$

holds. Finally, we obtain

$$\begin{aligned} V_{l+1}(x) &= (1 - \lambda_l(x))V_l(x) + \lambda_l(x)\Gamma_{l+1}(x) \\ &\leq (1 - \lambda_l(x))V_l(x) + \lambda_l(x)\Gamma_l(x) \\ &\leq (1 - \lambda_l(x))V_l(x) + \lambda_l(x)V_l(x) = V_l(x). \end{aligned}$$

Thus, the induction step is proved and (8) holds. ■

With the monotonicity of the approximate value function, we can show that each control law π_i renders the system asymptotically stable, as shown in the next theorem.

Theorem 3: Let Assumption 1 and 2 hold. If the local VI (6) is initialized with the value function of an admissible policy, then each associated control law $\pi_i, \ i \in \mathbb{Z}_0$, stabilizes system (1), and $\mathcal{B}^i_{\overline{r}_i}$ is in the region of attraction of π_i, where $\mathcal{B}^i_r \coloneqq \{x \in \mathbb{R}^n : V_i(x) \leq r\}$ and $\overline{r}_i$ is the largest r such that $\mathcal{B}^i_r \subset \Omega$. Specifically, for all $i \in \mathbb{Z}_0$ it holds that

$$V_i(f(x, \pi_i(x))) - V_i(x) \leq -l(x, \pi_i(x)), \quad \forall x \in \mathcal{B}^i_{\overline{r}_i}. \tag{10}$$

Proof: We use V_i as a Lyapunov function. For that we note that the following bounds hold

$$l(x,0) \leq V_i(x) \leq V_0(x), \ \forall i \in \mathbb{Z}_1, x \in \Omega, \tag{11}$$

where the lower bound holds because for any $V_{i-1}(x)$ we have for $x \in \Omega$

$$\begin{aligned} V_i(x) &= (1 - \lambda_{i-1}(x))V_{i-1}(x) + \lambda_{i-1}(x)\Gamma_i(x) \\ &\geq (1 - \lambda_{i-1}(x))V_{i-1}(x) + \lambda_{i-1}(x)l(x,0) \geq l(x,0). \end{aligned}$$

Furthermore, note that $V_0(x)$ is upper bounded by a function $\overline{\alpha}_{V_0} \in \mathcal{K}_\infty$, because it is the value function of

the stable policy π. We conclude that V_i is a Lyapunov function candidate. We show the decay rate of V_i by using induction. We assume that $V_{i-1}(f(x,\pi_{i-1}(x)))-V_{i-1}(x) \leq -l(x,\pi_{i-1}(x))$, $\forall x \in \Omega$. For $x \in \mathcal{L}_{i-1}$, we get from (6)

$$\begin{aligned} V_i(x) &= (1-\lambda_{i-1}(x))V_{i-1}(x) + \lambda_{i-1}(x)\Gamma_i(x) \\ &\overset{(8),(9)}{\geq} (1-\lambda_{i-1}(x))V_i(x) + \lambda_{i-1}(x)\Gamma_{i+1}(x) \\ &= (1-\lambda_{i-1}(x))V_i(x) \\ &\quad + \lambda_{i-1}(x)(l(x,\pi_i(x)) + V_i(f(x,\pi_i(x)))). \end{aligned}$$

Reorganizing the inequality such that only the stage cost is on the right-hand side, we obtain

$$\lambda_{i-1}(x)(V_i(x) - V_i(f(x,\pi_i(x)))) \geq \lambda_{i-1}(x) l(x,\pi_i(x)).$$

Thus, for $x \in \mathcal{L}_{i-1}$, (10) holds. For $x \in \Omega \setminus \mathcal{L}_{i-1}$, $V_i(x) = V_{i-1}(x)$, from (7), we obtain $\pi_i(x) = \pi_{i-1}(x)$. Then

$$\begin{aligned} V_i(f(x,\pi_i(x))) - V_i(x) &= V_i(f(x,\pi_{i-1}(x))) - V_{i-1}(x) \\ &\leq V_{i-1}(f(x,\pi_{i-1}(x))) - V_{i-1}(x) \\ &\leq -l(x,\pi_{i-1}(x)) = -l(x,\pi_i(x)), \end{aligned}$$

where the first inequality holds, because $f(x,\pi_{i-1}(x)) \in \Omega$ and (8) hold for any x in a sublevel set of V_{i-1} in Ω. Notice that from (11) it holds that $\mathcal{B}^0_{\overline{r}_0} \supseteq \mathcal{B}^i_{\overline{r}_i}$, $\forall i \in \mathbb{Z}_1$. For the base case we see that $V_0(x) = V_\pi(x)$. From (4) the base case holds. Thus, we have established the decay rate of the Lyapunov function by induction and conclude that a trajectory from a point in any sublevel set of V_i in Ω converges to the origin under π_i. We also conclude that the largest sublevel of V_i in Ω is in the region of attraction. ■

Remark 3: In comparison to Theorem 2, we do not show convergence of local VI, as we are only interested in the properties of the approximated function *during* iterations. Thus, we do not require convergence of the approximation.

In the next section we embed the approximated value function in an MPC framework as a means to improve the suboptimality of the closed-loop system. Furthermore, we give stability and recursive feasibility guarantees.

B. VI-based Model Predictive Control

The proposed predictive controller is given by the following optimization problem

$$V_{\text{MPC},N}(x,k) \coloneqq \tag{12a}$$

$$\min_{\overline{u}(\cdot;x,k)} \sum_{j=0}^{N-1} l(\overline{x}(j;x,k),\overline{u}(j;x,k)) + V_k(\overline{x}(N;x,k)) \tag{12b}$$

$$\text{s.t. } \overline{x}(j+1;x,k) = f(\overline{x}(j;x,k),\overline{u}(j;x,k)),\ \overline{x}(0;x,k) = x \tag{12c}$$

$$(\overline{x}(j;x,k),\overline{u}(j;x,k)) \in \mathbb{D}, \quad \forall j \in \{0,\dots,N-1\}, \tag{12d}$$

where V_k is the current value function approximation as defined in (6). Notice that the terminal cost varies with time. The solution of the optimization problem is denoted by $\overline{u}^*(\cdot;x,k)$ with associated state trajectory $\overline{x}^*(\cdot;x,k)$. At each time step the optimization problem (12) is solved using the current state measurement $x = x(k)$ and the input $u(k) = \overline{u}(0;x(k),k)$ is applied to the system.

C. Stability results

We require the subsequent assumptions to derive an appropriate horizon length to guarantee stability.

Assumption 3: There exists a constant $\varepsilon > 0$ such that the set $\mathcal{B}_\varepsilon \coloneqq \{x \in \mathbb{R}^n \,|l(x,0) \leq \varepsilon\}$ is contained in Ω and $(x,\pi(x)) \in \mathbb{D}$, and $x \in \mathcal{B}^0_{\overline{r}}$, $\forall x \in \mathcal{B}_\varepsilon$.

Assumption 4: There exists a constant $\gamma > 0$ such that $V_{\text{MPC},N}(x,0) \leq \gamma l(x,0), \forall x \in \mathcal{B}_\varepsilon$.

Note that this implies that $V_{\text{MPC},N}(x,k) \leq \gamma l(x,0), \forall x \in \mathcal{B}_\varepsilon$, $\forall k \in \mathbb{Z}_0$, as (8) holds.

Remark 4: Assumption 3 implies that the controller π fulfills the system constraints locally around the origin and the set $\mathcal{B}_\epsilon$ is completely contained in $\mathcal{B}^0_{\overline{r}}$, in order to use Theorem 3. Such a controller can be found, e.g., in case the linearization of the system is stabilizable, by designing an LQR controller based on the system linearization and considering a sufficiently small neighborhood around the origin. The set inclusion condition can be fulfilled with a small enough ε. Assumption 4 requires the system to be controllable *sufficiently* fast to the origin. The exponential cost controllabilty in [9] and [5, Section 6.2] can be expressed as required in the assumption.

To guarantee stability of the closed-loop system and recursive feasibility, we determine a sufficiently long prediction horizon. The terminal predicted state is required to lie in the set $\mathcal{B}_\varepsilon$ to exploit the properties of the learned terminal cost. To do this, consider the following lemma, adapted from [15].

Lemma 3: (Terminal state) Let Assumptions 1, 3, and 4 hold. Then, for any $\overline{V} > 0$ there exists $N_\Omega \in \mathbb{Z}_1$ such that for all $N \in \mathbb{Z}_1$, $N \geq N_\Omega$, any $k \in \mathbb{Z}_0$ and any $x \in \mathbb{X}_{\overline{V}} \coloneqq \{y \in \mathbb{X} | V_{\text{MPC},N}(y,k) \leq \overline{V}\}$, it holds that $\overline{x}^*(N;x,k) \in \mathcal{B}_\varepsilon$. Additionally, $V(N;x,k) \leq \rho_\gamma^{N-N_0} \min\{\gamma l(0,\overline{u}^*(0;x,k)), \underline{\gamma}\varepsilon\}$ holds, where $N_0 = \left\lceil \max\left\{0, \frac{\overline{V}-\gamma\varepsilon}{\varepsilon}\right\}\right\rceil$ and $V(k;x,k) = V_{\text{MPC},N-k}(\overline{x}^*(k;x,k))$, with $\rho_\gamma = \frac{\gamma-1}{\gamma}$ and $\underline{\gamma} = \min\{\gamma, \overline{V}/\varepsilon\}$.

Proof: As shown in [9, Theorem 5], by selecting

$$N \geq N_\Omega = N_0 + \left\lceil \frac{\max\{\log(\underline{\gamma}),0\}}{\log(\gamma) - \log(\gamma-1)} \right\rceil, \tag{13}$$

we obtain $V(N;x,k) \leq \varepsilon$. Then $\varepsilon \geq V_k(\overline{x}(N;x,k)) \geq l(\overline{x}(N;x,k),0)$, where the last inequality holds by (11). Thus, from Assumption 3 we obtain $\overline{x}(N;x,k) \in \Omega$. ■

The following theorem presents the main stability result.

Theorem 4: Let Assumptions 1–4 hold. Furthermore, let VI be initialized by using an admissible policy $\pi(\cdot)$. Then, there exists $N_{\overline{V}} \in \mathbb{Z}_1$ such that for any $N \geq N_{\overline{V}}$ and any $x_0 \in \mathbb{X}_{\overline{V}} \coloneqq \{y \in \mathbb{R}^n \mid V_{\text{MPC},N}(y,0) \leq \overline{V}\}$, the predictive control problem (12) is feasible for all $k \in \mathbb{Z}_0$, the constraints are satisfied, and the origin is asymptotically stable for the resulting closed loop.

Proof: We show the claim by using $V_{\text{MPC},N}$ as a Lyapunov function. First, notice that

$$l(x,0) \leq V_{\text{MPC},N}(x,k) \leq \gamma l(x,0), \quad \forall k \in \mathbb{Z}_0, \tag{14}$$

with $\gamma := \max\{\gamma, \overline{V}/\varepsilon\}$, as shown in [9, Theorem 5]. The upper bound holds using a case distinction, whether $x \in \mathcal{B}_\varepsilon$ or not. Define $\Delta V_{\text{MPC},N} := V_{\text{MPC},N}(x(k+1), k+1) - V_{\text{MPC},N}(x(k), k)$. Consider an input sequence $\hat{u}(\cdot; k)$ such that $\hat{u}(j; k) = \overline{u}^*(j+1; x(k), k)$ for $j \in \{0, \dots, N-2\}$ and $\hat{u}(N-1; k) = \pi_k(\hat{x}(N-1; x(k), k))$, where $\hat{x}(j; k)$ is the associated system trajectory. Then, with $N \geq N_\Omega$ and $x(k) \in \mathbb{X}_{\overline{V}}$, from Lemma 3, $\hat{x}(N-1; k) = \overline{x}^*(N; x(k), k) \in \mathcal{B}^0_{\overline{r}_0}$. Because of this, $\pi_k(\hat{x}(N-1; k))$ is a feasible control input and $\hat{u}(\cdot; k)$ is a feasible input sequence for the problem (12) at time step $k+1$, thus

$$
\begin{aligned}
&\Delta V_{\text{MPC},N} \leq -l(x(k), u(k)) \\
&+ l(\overline{x}^*(N; x(k), k), \hat{u}(N-1; k)) \\
&+ V_{k+1}(\hat{x}(N; k)) - V_k(\overline{x}^*(N; x(k), k)) \\
&= -l(x(k), u(k)) \\
&+ l(\overline{x}^*(N; x(k), k), \pi_k(\overline{x}^*(N; x(k), k))) \\
&+ V_{k+1}(f(\overline{x}^*(N; x(k), k), \pi_k(\overline{x}^*(N; x(k), k)))) \\
&- V_k(\overline{x}^*(N; x(k), k)) \overset{(8),(10)}{\leq} -l(x(k), u(k)).
\end{aligned}
$$

Then, the optimal control problem with $x_0 \in \mathbb{X}_{\overline{V}}$ is feasible for all $k \in \mathbb{Z}_0$ and $V_{\text{MPC},N}$ is a Lyapunov function for the closed-loop system. Thus, the origin is asymptotically stable for all $x_0 \in \mathbb{X}_{\overline{V}}$. ∎

IV. Simulation Study

We consider the orbital maneuver problem which has been addressed using ADP in [8] and [1]. The state of the system is chosen as $x(k) = [X(k), Y(k), X_t(k), Y_t(k)]^\top$, where $[X, Y]^\top$ describes the position of the spacecraft measured from the orbital frame in the destination orbit and $[X_t, Y_t]^\top$ its velocity. The discretized dynamics of the system are given by

$$
\begin{aligned}
x(k+1) &= x(k) + \\
&\Delta t \begin{bmatrix} x_3(k) \\ x_4(k) \\ 2x_4(k) - (1 + x_1(k)) \left(\frac{1}{r(k)^3} - 1 \right) \\ -2x_3(k) - x_2(k) \left(\frac{1}{r(k)^3} - 1 \right) \end{bmatrix} + \Delta t \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} u(k)
\end{aligned}
$$

with $r(k) = \sqrt{(1 + x_1(k))^2 + x_2(k)^2}$. We consider constraints $\mathbb{X} = [-0.5, 0.5]^4$, $\mathbb{U} = [-2, 2]^2$ and use a sampling time of $\Delta t = 0.05$. As weighting matrices we select $Q = \text{diag}(50, 50, 50, 50)$ and $R = \text{diag}(1, 1)$. All the calculations and simulations were done in MATLAB and *fmincon* was used to solve nonlinear optimization problems. We select the approximator as a linear combination of basis functions as done in [6] and [1],

$$
V_i(x) = w_i^T [(x \otimes x)^\top,\ (x \otimes x \otimes x)^\top]^\top.
$$

The local VI is initialized using a value function of an admissible policy obtained using *global* value iteration over the domain $\Omega = [-0.23,\ 0.23]^4$ and using 5000 samples from a uniform distribution and the previously introduced function approximator, however, without achieving convergence. The stability of the control law is verified through samples using the results in [15], as the convergence, measured with the constant $c_\delta = 0.55$ and the approximation error with $c_\varepsilon = 0.38$ fulfill the condition $c_\varepsilon + c_\delta < 1$. This results in $\varepsilon \approx 2.7$ and $\gamma \approx 28.5$. Using these values in (13) we obtain a required horizon of 236. This horizon is conservative, as the system converges to the origin for much shorter horizons. This was also seen in [1] and [15]. The effect of the different choices of the terminal cost is best seen using a shorter horizon that also stabilizes the system. Thus, we use a horizon of 10 in our simulation study. We evaluate initial conditions of the form $x_{0,j} = [x_{1,j}, x_{2,j}, -2, 2]$ and use 100 simulation steps. The system is simulated with the MPC controller using online VI, offline VI and without terminal cost. We select $\mathcal{L}_k = \{x \in \Omega \mid \|x - \overline{x}^*(N; x(k-1))\| \leq d_N\}$, and set $d_N = 0.144$. We also use 5000 random samples over Ω for the local update, which are filtered according to each $\mathcal{L}_k$. For $x \in \mathcal{L}_k$, the learning rate is selected as a Gaussian type function [17]

$$
\lambda_k(x) = \exp\left(- \|x - \overline{x}^*(N; x(k-1))\|^2 / (2\sigma^2) \right).
$$

The constant σ is set to 0.067, so that the learning rate is almost zero at a distance of 0.15 from the predicted terminal state and larger than 0.8 for a distance of 0.04 or less. In other words, we update the approximation of the value function in the vicinity of the predicted terminal state at the previous time step. The reason behind this is to use the last predicted terminal state as a proxy for the unknown current predicted terminal state, as the terminal cost can be regarded as the cost tail of an infinite-horizon problem.

Figure 1 shows a comparison of the closed loop performance for a predictive controller with the learned terminal cost and one without it. The performance improvement is measured as the reduction in the closed-loop cost, taking the controller without terminal cost as the starting point. It can be seen that there was significant improvement over most of the inspected initial conditions. The highest improvement is seen for initial conditions near the upper left corner of the grid, while the lowest was seen for the points around the origin.

A similar result is seen in Figure 1, where the performance of the online VI-based controller was compared with the performance of controllers with a value function approximated offline. In this case, the performance improvement is zero if the resulting performance is the same as that of the controller with the immature approximation, used to start the local VI, and 100 percent if it achieves the performance of the controller with a mature one, obtained by continuing offline VI for the immature approximation. The greatest performance improvement is seen for the points farthest away from the origin, and the lowest for a neighborhood around the origin. This behavior may be due to the fact that the online scheme can perform more iterations for points farther away from the origin, and thus obtain a better approximation. However, the dynamics of the system and the constraints also play a role. It must be noted that the absolute difference between the performance of both controllers with constant

terminal cost is very small, with a maximum value of 0.12, while the performance of the online VI-based controller oscillates between 460 and 15. Regarding the computation time, the *average* time per iteration for global VI was 34 seconds, while the local VI needed 1.84 seconds on the computing server Gigant of the Chemnitz University of Technology. Thus, using local VI significantly reduces the time needed.

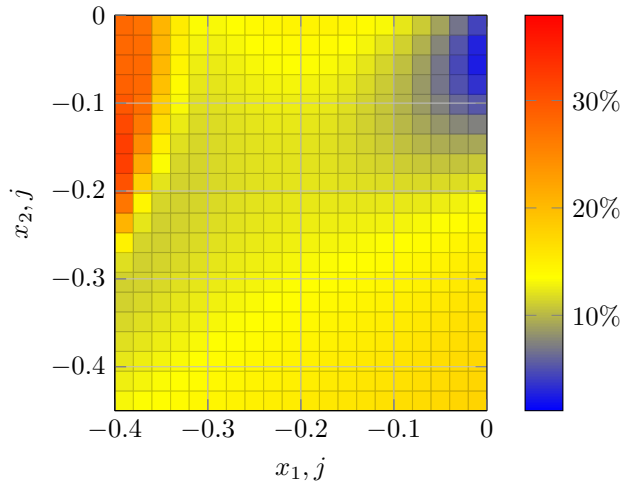

Fig. 1. Performance improvement, online VI and no terminal cost.

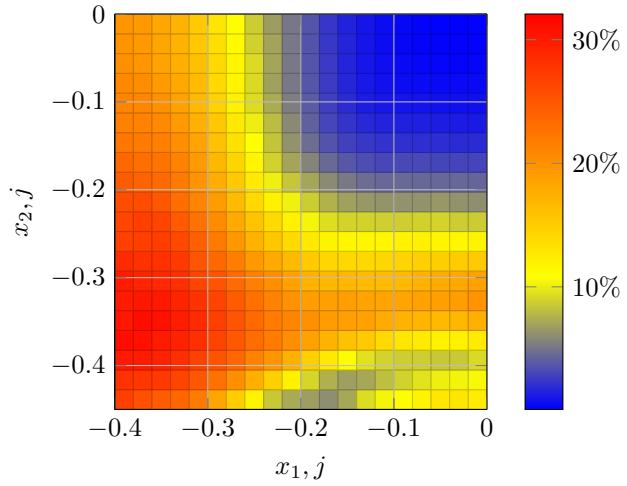

Fig. 2. Performance improvement, online VI and offline VI.

V. Conclusion

We propose an MPC controller that uses an online approximation of an infinite-horizon optimal value function as the terminal cost, obtained using online VI. First, we showed that using a local VI update, controlled using a state-dependent learning rate, we can preserve the stability of the associated controller for all iterations. This enables us to use a long-enough prediction horizon, which depends on the desired region of attraction and the design parameters of the MPC controller, to give stability guarantees. The local VI step can be considered as extending the prediction horizon of the controller by one step at each time step, however, with a relatively constant computation time. Even though the proposed controller provides a more suboptimal closed-loop performance in comparison to a controller based on *global* VI, the computational demand can be significantly reduced by using the learning rate and the sets that control the update. Furthermore, the proposed controller may be extended with a robust or adaptive framework, for cases where information about the system is only available for a subset of the state space, and controllers such as the one presented in [15], are not applicable. Even though the resulting stabilizing horizon is conservative, a common shortcoming of this type of analyses, a short prediction horizon could be used in conjunction with a terminal constraint. Stability and recursive feasibility of the controller are still guaranteed, provided the optimization problem is initially feasible. Future research may improve on the required stabilizing horizon. Also, the influence of the approximation error on the properties of the learned value function can be analyzed, as the approximators used do not provide an exact solution of the iteration.

References

[1] Lukas Beckenbach and Stefan Streif. Approximate infinite-horizon predictive control. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 3711–3717. IEEE, 2022.

[2] Dimitri P Bertsekas. Value and policy iterations in optimal control and adaptive dynamic programming. *IEEE transactions on neural networks and learning systems*, 28(3):500–509, 2015.

[3] Hong Chen and Frank Allgöwer. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica*, 34(10):1205–1217, 1998.

[4] Giuseppe De Nicolao, Lalo Magni, and Riccardo Scattolini. Stabilizing receding-horizon control of nonlinear time-varying systems. *IEEE Transactions on Automatic Control*, 43(7):1030–1036, 1998.

[5] Lars Grüne and Jürgen Pannek. *Nonlinear model predictive control*. Springer, 2017.

[6] Ali Heydari. Theoretical and numerical analysis of approximate dynamic programming with approximation errors. *Journal of Guidance, Control, and Dynamics*, 39(2):301–311, 2016.

[7] Ali Heydari. Stability analysis of optimal adaptive control under value iteration using a stabilizing initial policy. *IEEE Transactions on Neural Networks and Learning Systems*, 29(9):4522–4527, 2018.

[8] Ali Heydari and SN Balakrishnan. Adaptive critic-based solution to an orbital rendezvous problem. *Journal of Guidance, Control, and Dynamics*, 37(1):344–350, 2014.

[9] Johannes Köhler and Frank Allgöwer. Stability and performance in MPC using a finite-tail cost. *IFAC-PapersOnLine*, 54(6):166–171, 2021.

[10] Frank L Lewis and Derong Liu. *Reinforcement learning and approximate dynamic programming for feedback control*. John Wiley & Sons, 2013.

[11] Bo Lincoln and Anders Rantzer. Relaxing dynamic programming. *IEEE Transactions on Automatic Control*, 51(8):1249–1260, 2006.

[12] Derong Liu and Qinglai Wei. Policy iteration adaptive dynamic programming algorithm for discrete-time nonlinear systems. *IEEE Transactions on Neural Networks and Learning Systems*, 25(3):621–634, 2013.

[13] Lalo Magni, Giuseppe De Nicolao, Lorenza Magnani, and Riccardo Scattolini. A stabilizing model-based predictive control algorithm for nonlinear systems. *Automatica*, 37(9):1351–1362, 2001.

[14] David Q. Mayne, James B. Rawlings, Christopher V Rao, and Pierre OM Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[15] Francisco Moreno-Mora, Lukas Beckenbach, and Stefan Streif. Predictive control with learning-based terminal costs using approximate value iteration. *IFAC-PapersOnLine*, 56(2):3874–3879, 2023. 22nd IFAC World Congress.

[16] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[17] Qinglai Wei, Frank L. Lewis, Derong Liu, Ruizhuo Song, and Hanquan Lin. Discrete-time local value iteration adaptive dynamic programming: Convergence analysis. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48(6):875–891, 2018.