Learning near-optimal broadcasting intervals in decentralized multi-agent systems using online least-square policy iteration

Ivana Palunko1 | Domagoj Tolić2 | Vicko Prkačin1

1 LARIAT-Laboratory for Intelligent Autonomous Systems, University of Dubrovnik, Dubrovnik, Croatia
2 LARIAT-Laboratory for Intelligent Autonomous Systems, RIT Croatia, Dubrovnik, Croatia

Abstract
Here, agents learn how often to exchange information with neighbours in cooperative multi-agent systems (MASs) such that their linear quadratic regulator (LQR)-like performance indices are minimized. The investigated LQR-like cost functions capture trade-offs between the energy consumption of each agent and MAS local control performance in the presence of exogenous disturbances, delayed and noisy data. Agent energy consumption is critical for prolonging the MAS mission and is composed of both control (e.g. acceleration, velocity) and communication efforts. Taking provably stabilizing upper bounds on broadcasting intervals as optimization constraints, an online off-policy model-free learning algorithm based on least square policy iteration (LSPI) to minimize the cost function of each agent is employed. Consequently, the obtained broadcasting intervals adapt to the most recent information (e.g. delayed and noisy agents’ inputs and/or outputs) received from neighbours whilst provably stabilize the MAS. Chebyshev polynomials are utilized as the approximator in the LSPI whereas Kalman filtering handles sampled, corrupted, and delayed data. Subsequently, convergence and near-optimality of our LSPI scheme are inspected. The proposed methodology is verified experimentally using an inexpensive motion capture system and nano quadrotors.

1 INTRODUCTION
Characterized by local interactions among neighbours, decentralized control of multi-agent systems (MASs) is one of the most active research areas in the last two decades [1–7]. As discussed in the aforementioned references, information exchange among neighbours is instrumental for MAS coordination, especially in the presence of exogenous disturbances, delayed and noisy measurements. However, even for a single control loop (i.e. for a single agent), it is well known that information exchange at “slow enough” [8] or “fast but not too fast” [9, Remark 5] rates is required for stabilization in the presence of measurement noise. This data exchange rate dependency is also evident from the $L_2$-gain and bias terms in [10, 11]. Besides attaining merely closed-loop stability, one is usually interested in achieving satisfactory control system performance [12–18]. These performance-oriented works consider intermittent feedback [19] and variants of the famed linear quadratic regulator (LQR) optimal control problem [20] are employed to account for data acquisition, processing and/or communication costs. Recall that the classical LQR problem involves merely the control (e.g. acceleration, velocity) cost. LQR-like performance indices, which address information-related costs next to the control cost, in MAS settings are investigated in [10, 21] and the present work.

Moreover, intermittent information/feedback and networked control systems (NCSs) are of interest in light of real-life control system implementations [12, 19]. Namely, owing to digital technology and limited communication channel capacities (i.e. bandwidth), continuous or periodic data flows among neighbours are often not achievable, but instead the data among agents is exchanged intermittently. In addition, realistic communication channels introduce delays, distort the transmitted information and give rise to packet dropouts. In consequence of these network-induced phenomena, the MAS control performance impairs and the MASs can even become unstable. At the same
time, agents’ sensing and broadcasting units consume energy while in operation. Since agents typically have scarce energy supplies at disposal (e.g. batteries), it may erroneously seem that sensing and broadcasting should be executed as sparsely as possible in order to prolong the MAS mission (i.e. agents’ lifetime), which is the sole driving force in some of the related works (see the discussions in [10, 14, 22] and the references therein). For instance, compared to scenarios with “frequent” sampling and broadcasting, “infrequent” data exchange results in greater settling times, that is, in slower convergence to a vicinity of MAS equilibria impairing disturbance rejection [22]. In turn, “infrequent” data exchange may exhaust the energy supplies more than the “frequent” counterpart due to greater control efforts in the long run. This observation underlies the present work and its predecessors [10, 14, 22].

Given some low-level and group cooperation controllers, this article quantifies the repercussions of intermittent feedback in the group cooperation controllers on the MAS control performance and MAS lifetime. To that end, a continuous-time LQR-like optimization problem is posed for each agent involving only locally available data. Each cost function captures the local control performance versus agent lifetime trade-offs. In theory, dynamic programming (DP) solves a wide spectrum of optimization problems providing an optimal solution [23, 24], including the LQR-like optimization problems considered herein. In practice, straightforward implementations of DP algorithms are deemed computationally intractable for most applications due to the infamous curse of dimensionality [25]. Therefore the need for efficient model-based approximate dynamic programming (ADP) or model-free reinforcement learning (RL) methods (refer to [25, 26] and [27–29], respectively). The RL method utilized herein is the online off-policy model-free algorithm based on leastsquare policy iteration (LSPI) reported in [30, 31]. The off-policy feature of the algorithm utilized herein was inherited from [32]. The aforementioned local DP problems are coupled, which in turn yields non-autonomous underlying dynamics so that the corresponding costs-to-go are non-stationary [33]. In addition to availability of often imprecise mathematical models of agent dynamics, the distribution of exogenous disturbances is usually unknown or unpredictable (e.g. winds, sea currents, waves) as well as the behaviour of neighbouring agents due to the influence of other agents, disturbances and noise. Hence the need for an online model-free RL method. Moreover, online RL methods do not require the training phase (unlike offline RL methods), which may be a prohibitive factor in practice. Our LSPI approximation architecture for both continuous states and actions is based on Chebyshev polynomials. In addition, we use Kalman filtering (KF) to alleviate the problem of delayed, sampled and noisy data [34], [23], Chapter 4.

In comparison with its precursor [10], the present article provides sufficient convergence and near-optimality guarantees and experimental verification of the online LSPI. In our framework, we first compute stabilizing upper bounds for intervals between two consecutive broadcasting instants of each individual agent (utilizing agents’ approximate linear time invariant (LTI) dynamical models), thereby giving rise to asynchronous communication. Afterwards, these stabilizing upper bounds are exploited as optimization constraints in the online LSPI aiming for optimality. To the best of our knowledge, the optimal MAS intermittent feedback problem is still an open problem. For example, in comparison with this work, the authors in [33] and [35] employ different RL approaches and aim for different optimization goals. The work in [36] considers a cooperative performance index, rather than individual cost functions as we do, and imposes restrictions on graph edge weights to ensure that all agents play the same game and that learning converges whereas [37] seeks the LQR control law for a single agent in deterministic settings with transmission instants given by an external event-triggered mechanism. Likewise, [38] designs/learns control laws whereas we design/learn broadcasting intervals for already designed controllers. Furthermore, [39] requires at least one pinning node, which serves as the command generator, and considers identical agents with state feedback in the absence of delays, noise and disturbances. Altogether, the main contributions of this article are threefold: (a) RL algorithm convergence conditions; (b) near-optimality bounds; and (c) experimental verification employing affordable off-the-shelf localization system and nano quadrotors as MAS agents.

The remainder of the article is organized as follows. We finish off this section with basic notation and terminology. Section 2 formulates the optimal LQR-like intermittent feedback problem for MASs. In Section 3, we bring together LSPI and \( \mathcal{L}_\infty \)-stability with bias of MASs with delay dynamics in order to solve the LQR-like problem of interest. Section 4 is reserved for convergence and suboptimality analyses. A low-cost experimental validation is delineated in Section 5. Conclusions and future research avenues are in Section 6.

**Notation.** We use \( (x, y) = \begin{bmatrix} x^T & y^T \end{bmatrix}^T \). The dimension of a vector \( x \) is \( n \), whereas \( \| \cdot \| \) denotes the Euclidean norm of a vector. If the argument of \( \| \cdot \| \) is a matrix, then it denotes the induced matrix 2-norm. The kernel of a matrix \( A \) is \( \text{Ker}(A) \). The set cardinality is denoted \( | \cdot | \). An \( n \times n \) identity matrix is \( I_n \).

For the sake of brevity, we write “w.r.t.” instead of “with respect to”. Let

\[
\| f(a, b) \|_{p, B} := \left( \frac{1}{|b-a|} \int_{[a,b]} \| f(\gamma) \|_{p, B}^p \, d\gamma \right)^{1/p},
\]

where \( \| f(\gamma) \|_B := \inf_{\delta \in B} \| f(\gamma) - \delta \| \) and \( p \in [1, \infty] \), denote the \( \mathcal{L}_p \)-norm w.r.t. a set \( B \subset \mathbb{R}^n \) of a Lebesgue measurable function \( f : \mathbb{R} \to \mathbb{R}^n \) restricted to the interval \([a, b] \subset \mathbb{R} \).
A directed graph is a pair $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, ..., v_N\}$ is a non-empty set of nodes, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges. When the edge $(i, j)$, $i \neq j$, belongs to $\mathcal{E}$, it means that there is an information flow from the node $i$ to the node $j$. The set of neighbours of the node $v_i$ is $\mathcal{N}_i = \{j \in \mathcal{V}: (i, j) \in \mathcal{E}\}$. The graph Laplacian matrix $L \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ is defined as

$$L = [l_{ij}], \quad l_{ij} = \begin{cases} -1, & j \in \mathcal{N}_i, \quad j = i \quad \text{.} \\ 0, & \text{otherwise} \end{cases}$$

2 PROBLEM STATEMENT

Consider $N$ heterogeneous linear agents given by

$$\dot{\xi}_i = A_i \xi_i + B_i u_i + \omega_i, \quad \xi_i = C_i \xi_i,$$

where $\xi_i \in \mathbb{R}^{n_i}$ is the state, $u_i \in \mathbb{R}^{n_u}$ is the input, $\xi_i \in \mathbb{R}^{n_i}$ is the output of the $i$th agent, $i \in \{1, 2, ..., N\}$, and $\omega_i \in \mathbb{R}^{n_\omega}$ reflects (often unknown and possibly stochastic) exogenous disturbances and/or modelling uncertainties. Matrices $A_i$, $B_i$ and $C_i$ are of appropriate dimensions. A common decentralized policy is

$$u_i(t) = -K_i \sum_{j \in \mathcal{N}_i} (\xi_j(t) - \hat{\xi}_j(t)), \quad \dot{\hat{\xi}}_j = A \xi_j + B \eta_j + \omega_j,$$

where $K_i$ is an $n_u \times n_\omega$ gain matrix [1, 2]. In real-life applications, the transmitted information about signals $\dot{\xi}_i(t)$ and $\dot{\hat{\xi}}_j(t)$ in (2) arrive at the receivers with some delay. For simplicity, suppose that all communication links introduce the same propagation delay $d \geq 0$ so that $\dot{\xi}_i(t)$ and $\dot{\hat{\xi}}_j(t)$ in (2) can be replaced with $\dot{\xi}_i(t - d)$ and $\dot{\hat{\xi}}_j(t - d)$, respectively.

Remark 1. Delay constancy, and hence delay uniformity (i.e. the delays in all communication channels are equal), can be enforced via time-stamping of data or protocols with guaranteed and known upper bounds on the time delay and appropriate buffering at the receiver ends (refer to [40] and the references therein). In case time-varying and/or non-uniform delays are inevitable, the present approach can readily be augmented with the ideas from [11] owing to linearity of dynamics (at the expense of more complex notation and mathematics).

Next, we define the following stack vectors $\xi := (\xi_1, ..., \xi_N)$, $\hat{\xi} := (\hat{\xi}_1, ..., \hat{\xi}_N)$ and $\omega := (\omega_1, ..., \omega_N)$. Utilizing the Laplacian matrix $L$ of the underlying communication graph $G$, the closed-loop dynamic equation becomes

$$\dot{\xi}(t) = A \xi(t) + A^{cl} \hat{\xi}(t - d) + \omega(t), \quad \xi = C^{cl} \xi,$$

where $A^{cl}$ are matrix blocks whilst $\text{diag}(\cdot, ... , \cdot)$ indicates a diagonal matrix.

Next to being delayed, signals $\xi_i(t)$ and $\hat{\xi}_j(t)$ forwarded to the controllers (2) are typically sampled and noisy versions of the neighbours’ outputs. We denote these signals $\dot{\xi}_i(t)$ and $\dot{\hat{\xi}}_j(t)$, respectively. We point out that our approach does not require the zero-order-hold (ZOH) sampling (i.e. $\dot{\xi}_i(t) \equiv 0_{n_\xi}$ and $\dot{\hat{\xi}}_j(t) \equiv 0_{n_\xi}$ in between consecutive transmissions) but model-based estimators may be employed in order to extend the stabilizing broadcasting intervals [11] as demonstrated in Section 5. In addition to broadcasting the sampled versions of $\xi_i$’s at time instants belonging to $\mathcal{T}$ (see Figure 1), the agents may broadcast the sampled versions of $u_i$’s as well so that state estimates built upon both agents’ inputs and output can be provided to LSPI (see Section 3.2). If agents’ inputs are not available (e.g. non-cooperative MASs), one can readily employ input-free estimation schemes at the expense of increased estimation errors [41]. Noise related to $\xi_i$ is $v_i(t) \in \mathbb{R}^{n_\xi}$ whilst the noise pertaining to $u_i$ is $u_i(t) \in \mathbb{R}^{n_u}$. We assume $v_i(t)$ and $u_i(t)$ are bounded for each $i \in \{1, ..., N\}$.

In what follows, we need to differentiate among the broadcasting instants in $\mathcal{T}$ pertaining to each agent. Consequently, let $\mathcal{T}_i' \subseteq \mathcal{T}$, $i \in \mathcal{N}$, label the broadcasting instants of the $i$th agent. The broadcasting intervals $\mathcal{T}_i'$’s are defined by $\mathcal{T}_i' \equiv \tau_i' - \mathcal{T}_i' - 1$, $i \in \mathcal{N}$, with the convention $\tau_i' \equiv 0$ for each $j \in \{1, ..., N\}$. We allow broadcasting instants of different agents to coincide. Also, agents’ broadcasting instants are not orchestrated by some scheduling protocol. In other words, communication between agents is asynchronous. Furthermore, let us define local discrepancy vectors $\nu_i \in \{1, ..., N\}$, as follows: $\nu_i \equiv (\nu_{i1}, ..., \nu_{iN})$, where $\nu_i \in \mathcal{N}_i$.

According to [1–7], a significant portion of MASs (1) and (2) is designed to steer $\xi$ towards $B := \text{Ker}(A^{cl} + A^{cl}d)$ in an effort to attain $||\nu_i(t)|| \to 0$ as $t \to \infty$ for all $i \in \{1, ..., N\}$. In this article, we use $\nu_i$’s to measure the MAS local performance while the energy effort comprises both control and broadcasting/communication efforts. We are now ready to state the main problem solved herein.
Problem 1. For each $j \in \{1, \ldots, N\}$, minimize the following cost function

$$
\mathbb{E} \sum_{i=1}^{\infty} (\gamma_j)^i \left[ \int_{t_{j,i}}^{t_{j,i+1}} (x_j^T P_j x_j + u_j^T R_j u_j) \, dt + S_j \right]
$$

for the $j$th agent of MAS (1) and (2) over all sampling policies $\tau_j^i$ and for all initial conditions $x_j(t_0) \in \mathbb{R}^{n_j}$.

Remark 2. The classical LQR problem is noise and disturbance free, assumes proportional state-feedback control laws and does not consider the information cost $S_j$ [20, 23]. Hence, the states (and consequently, inputs) converge to zero exponentially in the classical LQR problem allowing for $\gamma_j = 1$ in (5) (see [36, 42, 43] for more). We do not seek for a stabilizing controller per se (as the related works do), but rather for when to feed a given control law with up-to-date information. Even though we consider continuous-time dynamics (1) and (2), the rewards are obtained at discrete time instants belonging to $\mathcal{T}$ (as seen from (5)), which renders our optimization problem a combination of the continuous- and discrete-time LQR problems. These characteristics impede straightforward comparisons of our results with the existing ones.

3 | METHODOLOGY

Our framework is intentionally set up such that the actual approaches in Sections 3.1 and 3.2 are irrelevant. In other words, the LSPI pursuit for optimal $\tau_j^i$’s in Section 3.2 and computation of stabilizing upper bounds on $\tau_j^i$’s in Section 3.1 can be replaced with more appropriate/advanced methods (once...
available) to meet one’s needs. Owing to this observation and in order to focus on the novelties of this work, we deliberately leave out the actual method from Section 3.1 of this article and replace it with a discussion of challenges that a successful method needs to overcome in light of Problem 1. Our method of choice, reported in [10, Section 4.1], explicitly utilizes the (approximate) LTI dynamics (1) and (2). A potential extension of [10, Section 4.1] towards non-uniform or distributed delays and less conservative stabilizing upper bounds on \( \tau_j \)’s is found in [44]. However, [44] does not consider disturbances nor noise.

The goal of Section 3.1 is to achieve an offline computation of upper bounds \( \bar{\tau} \) on \( \tau_j \)’s, which can be thought of as robustness margins, are utilized as optimization constraints in the online LSPI procedure of Section 3.2 while obtaining near-optimal \( \tau_j \)’s in the sense of Problem 1. In other words, we are trading off optimality of Section 3.2 for stability of Section 3.1 as discussed in [29, Section 4.1]. Accordingly, our solutions of the optimal problems (5) provably yield control system stability, which is a perennial problem when employing RL or ADP (see [29] or any RL/ADP reference herein for more), in addition to (sub)optimality of broadcasting instants. Furthermore, we employ KF in Section 3.2 to handle partially observable DP problems. It is to be noted that the (approximate) models (1) and (2) are employed in Section 3.1 and only for KF in Section 3.2. In other words, the RL part of our approach is completely model free as motivated in Sections 1 and 5.

### 3.1 Stabilizing broadcasting intervals

The method utilized in this section needs to be applicable to MASs with general heterogeneous continuous-time linear agents (not merely to single- or double-integrator dynamics) with exogenous disturbances, asynchronous data exchange, output feedback and directed communication topologies without pinning nodes. Furthermore, we favour approaches that allow for communication delays greater than the sampling period (the so-called large delays). At the moment, only [10, Section 4.1] takes into account the above problem settings along with concurrent adverse effects of the realistic data exchange phenomena such as sampled, corrupted and delayed data as well as lossy communication channels. Consequently, even though there exist a number of works investigating time- or self-triggered control strategies for MASs (refer to [46] for a list of such approaches), no other approach is applicable to settings as general as those considered herein. However, when some of the settings are relaxed or as novel approaches become available, alternatives for computing stabilizing upper bounds on \( \tau_j \), \( j \in \{1, \ldots, N\} \), \( i \in \mathbb{N} \), may be employed as well.

The sufficient conditions in [10, Section 4.1] typically entail the existence of a directed spanning tree (see [2, 22]), but one can envision less challenging/appealing MAS scenarios in which this directed spanning tree requirement is superfluous. In addition, since we are not interested exclusively in a single multi-agent differential graphical game, [10, Section 4.1] imposes no requirements on graph edge weights found in [36]. Accordingly, subsets of our agents can effectively be involved in distinct multi-agent differential graphical games.

Lastly, [10, Section 4.1] allows for model-based estimation in order to increase stabilizing upper bounds on \( \tau_j \)’s as exemplified in Section 5. These model-based estimators are not to be confused with stochastic filters/estimators employed in Section 3.2. The present subsection does not utilize stochastic estimates (nor predictions based upon these estimates) for feeding the controllers owing to the lack of explicit guarantees on estimation error upper bounds. Effectively, \( \psi_j(t_i) \) or \( \psi_j(t_i), i \in \{1, \ldots, N\} \), may cease to be bounded with a pre-specified constant. In practice, stochastic filtering can safely be used in control laws after the initial estimation error overshoot vanishes.

### 3.2 Learning near-optimal broadcasting intervals

For notational convenience, let us drop the agent identifier \( j \) while solving (5). Recall that \( i \) LSPI algorithms are executed in parallel (one on each agent) asynchronously.

Adaptive optimal algorithms from the literature are typically based on solving the Bellman equation [47]. In the present work, we use an online off-policy LSPI, which interacts with the control system at time instants \( t_i \), where \( i \) is the iteration step, \( i \in \mathbb{N} \), through the following three variables \( x(t_i), \tau(t_i) \) and \( r(t_i) \) found in the first variable (i.e. the LSPI state) is the local discrepancy vector \( x(t_i) \in \mathcal{X} \), which represents the local agent knowledge at time \( t_i \) and, in the present work, is obtained via estimation (e.g. one KF at each agent for all its neighbours). The second variable is the LSPI action \( \tau(t_i) \in \mathcal{A} \). The third variable is the LSPI reward function \( r(t_i) \) defined in (5), which measures the local performance after taking the action \( \tau(t_i) \) in state \( x(t_i) \). In other words, we are solving the sequential decision-making problem (5) and LSPI aims to provide optimal actions \( \tau(t_i) \) for each state \( x(t_i) \).

Using policy iteration, policies \( b_k(x(t_i)) \) are evaluated by iteratively constructing an estimate of the LSPI state and action value function called a \( Q \)-function. The use of \( Q \)-functions alleviates the problem of computing the expectation in (5) as discussed in [25, Section 8.2]. Following [30], the LSPI state-action
approximate value function is defined by

$$
\hat{Q}(x(t_i), \tau(t_i)) = \Phi^T(x(t_i), \tau(t_i)) \xi_k, \quad (6)
$$

where

$$
\Phi(x(t_i), \tau(t_i)) = \psi(\tau(t_i)) \otimes \phi(x(t_i))
$$

is the Kronecker product of the basis function vectors $\psi(\tau(t_i))$ and $\phi(x(t_i))$ formed with Chebyshev polynomials while $\xi_k$ is the approximation parameter vector that is being learned. Observe that the Kronecker product renders the basis $\phi$ orthonormal provided that bases $\psi$ and $\phi$ are orthonormal. Clearly, since Chebyshev polynomials are defined on the interval $[-1, 1]$, the state space $\mathcal{X}$ needs to be compact. The set $\mathcal{A}$ is also compact and its upper bound is $\mathcal{T}$ computed in Section 3.1. The lower bound of $\mathcal{A}$ is a small positive constant $\tau > 0$ stemming from hardware limitations (see Section 5 for more). Hence, $\mathcal{A} = [\tau, \mathcal{T}]$. Owing to the compactness of $\mathcal{A}$ and $\mathcal{X}$, finiteness of $r(t_i)$ is obtained so that the cost function in Problem 1 is well defined. The control action $\tau(t_i) \in \mathcal{A}$ is given by

$$
\tau(t_i) = b_k(x(t_i)), \quad (7)
$$

where

$$
b_k(x(t_i)) = \begin{cases} u.r.a. \in \mathcal{A} & \text{every } \varepsilon \text{ iterations,} \\ b_k(x(t_i)) & \text{otherwise,} \end{cases} \quad (8)
$$

where “u.r.a.” stands for “uniformly chosen random action” and yields the algorithm’s exploration phase every $\varepsilon$ steps while $b_k(x(t_i))$ is the policy obtained according to

$$
b_k(x(t_i)) = \arg \min_n \hat{Q}(x(t_i), \tau(t_i)). \quad (9)
$$

Since LSPI tackles optimization problems, the algorithm needs to have an exploration phase in order to avoid local minima and to cope with non-stationary cost-to-go functions [28, 29, 33]. In Section 5, owing to excursions from their perceived stationary distributions, the disturbances and measurement noise also serve as probing signals (or exploration noise) that ensure persistence of excitation aiding the exploration [20, 23, Chapter 11]. The parameter vector $\xi_k$ is updated every $\kappa \geq 1$ steps from the projected Bellman equation for model-free policy iteration that, according to [32], can be written as

$$
\Gamma_0 \xi_k = \gamma \Lambda_0 \xi_k + \zeta_0, \quad (10)
$$

where $\gamma$ originates from (5) and

$$
\Gamma_0 = \beta_1 I, \quad \Lambda_0 = 0, \quad \zeta_0 = 0,
$$

$$
\Gamma_i = \Gamma_{i-1} + \Phi(x(t_i), \tau(t_i)) \Phi(x(t_i), \tau(t_i))^\top, \quad (11)
$$

where $\Gamma_i$, $\Lambda_i$ and $\zeta_i$ represent the policy evaluation mapping and projection operator of the Bellman equation [28] and are updated at every iteration step $i$. The vector $\xi_k$, updated via (10), is used to improve the $Q$-function estimate (6). Afterwards, new and improved policies (in the sense of Problem 1) are obtained from (9). For more details regarding the LSPI algorithm employed herein, refer to [30] and [32].

Remark 3 ($\mathcal{L}_p$-stability and stochastic inputs). KF requires the disturbances $\omega_i$’s to be Gaussian white processes. As the Gaussian white processes are not Lebesgue integrable, one cannot invoke [10, Theorem 1], needed in Section 3.1, at once. However, the Gaussian white process is an idealization convenient for a number of analytical purposes, but does not exist as a physical phenomenon (see [48, Chapter 6], [49, Chapter 10] and [50]). Instead, band-limited approximations/realizations are encountered in practice. Therefore, all sample paths of $\omega_i$’s in Section 5 possess finite $\mathcal{L}_p$-norms and [10, Theorem 1] readily applies.

Remark 4. The investigation regarding suboptimality of the obtained broadcasting intervals for each agent is found in Section 4. The suboptimality repercussions of the KF-LSPI and LSPI-LSPI interplay in the MAS setting is still unclear. The generality of our MAS settings (e.g. partially observable and coupled (individual not collective) DP problems with external disturbances) hinders a fruitful comprehensive investigation at this point in time.

Remark 5. The benefits of LSPI include implementation simplicity due to linear-in-parameter approximators, which in turn strips the algorithm down to a set of linear matrix equations (10). Accordingly, the principal numerical concern lies in the need for matrix inversion, which is a well-understood nuisance circumvented, among others, via the LU factorization [51]. The complexity of solving (10) for $\xi_k$ is at most $O(n_k^3)$, but can be reduced by using computationally efficient methods to $O(n_k^2)$ as discussed in [28, Chapter 5]. Clearly, one can always decrease the numerical complexity of LSPI by employing lower-order Chebyshev polynomials at the expense of greater suboptimality in Section 4.4 (while system stability is still guaranteed in light of Section 3.1). Furthermore, the approximation in LSPI focuses on the value-function representation, thereby eliminating the “actor” part in the actor–critic framework of related approximate policy-iteration algorithms (see [33] and the references therein). Consequently, LSPI eliminates one potential source of inaccuracy and decreases computational complexity with respect to the actor–critic framework. Since LSPI is a model-free algorithm, it does not need a model for policy iteration nor it needs to learn one. Additionally, LSPI is able to learn (sub)optimal policies using a small number of samples when
compared to conventional learning approaches such as the classical $Q$-learning. For a comprehensive discussion, refer to [32].

4 ONLINE LSPI—CONVERGENCE AND SUBOPTIMALITY

In this section, we discuss online LSPI convergence and suboptimality for each individual agent assuming the external disturbance properties change slower than the learning takes place. Recall that we are not interested in the MAS collective behaviour but in agents’ individual behaviours as visible from the optimization problem (5) assigned to every agent. For notational simplicity, this section uses $\cdot_k$ instead of $(t_k)$ found in other sections. In addition, this section treats more general problems than Problem 1 so that its results do not apply merely to Problem 1. For example, even though the actions $\tau_k$ in other sections are positive scalars, this section allows for non-scalar actions in this section are denoted $a_k$ instead of $\tau_k$. Also, in order to make the notation of this section more in line with the RL literature, we slightly abuse notation found in other sections.

4.1 Reinforcement learning preliminaries

The goal of RL is to solve a stochastic discrete-time optimal control problem typically formalized as a *Markov decision process* (MDP) $(\mathcal{X}, \mathcal{A}, f, \rho)$ [32], where $\mathcal{X} \subseteq \mathbb{R}^{n_x}$ is the state space of the process, $\mathcal{A} \subseteq \mathbb{R}^{n_a}$ is the action space, $f : \mathcal{X} \times \mathcal{A} \times \mathcal{X} \to [0, \infty)$ is the transition probability function of the process, that is, the probability of making a transition to state $x_{k+1}$ when taking action $a_k$ in state $x_k$, and $\rho : \mathcal{X} \times \mathcal{A} \times \mathcal{X} \to \mathbb{R}$ is the reward function

$$r_{k+1} = \rho(x_k, a_k, x_{k+1}).$$

An important property of MDPs is that, under appropriate conditions, there exists a deterministic optimal policy despite the fact that the transitions are stochastic [20, 23]. Hence, the agent chooses actions according to its (deterministic) policy $b : \mathcal{X} \to \mathcal{A}$ (cf. (7))

$$a_k = b(x_k).$$

In particular, the RL objective is to minimize the infinite-horizon expected discounted return (cf. (5))

$$R^b(x_0) = \mathbb{E}\left\{ \sum_{k=0}^{\infty} \gamma^k \rho(x_k, b(x_k), x_{k+1}) \right\},$$

where $\gamma \in [0, 1]$ is the discount factor and $x_{k+1}$ is drawn from the density $f(x_k, a_k, \cdot)$ at each step $k$. In other words, one wants to find an optimal policy $b^*$ that procures the minimal value function from any initial state $x_0$

$$V^*(x_0) := \min_b V^b(x_0), \quad \forall x_0.$$  

Any policy that attains the minima in this equation is optimal.

When solving undiscounted problems (i.e. $\gamma = 1$) with unbounded rewards, one must impose stability conditions to ensure that values are bounded and the problem is well posed. Thus, the discount factor is usually taken subunitary and rewards are assumed to be bounded, which makes the value functions well behaved irrespective of stability concerns. Herein, we first ensure stability in Section 3.1 so that the reward boundedness assumption is superfluous for well posedness of Problem 1. Furthermore, instead of directly using value functions, LSPI uses $Q$-functions $Q : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ that fix the initial action. The intuitive meaning of the $Q$-value $Q^b(x, a)$ is that of expected return when starting from state $x$, applying the first action $a$, and following $b$ thereafter. The reason for preferring $Q$-functions is simple: once $Q^*$ is available, an optimal (greedy) policy can be computed easily by selecting at each state an action with the smallest optimal $Q^*$ value (cf. (9)):

$$b^*(x) \in \arg \min_a Q^b(x, a).$$

When there are multiple minimizing actions, any of them is optimal. In contrast, the formula to compute $b^*$ from $V^*$ is more complicated and involves a model, which in our setting is unavailable. The state value functions can be expressed in terms of $Q$-functions as

$$V^b(x) = Q^b(x, b(x)), \quad V^* = \min_a Q^*(x, a) = Q^*(x, b^*(x)).$$

4.2 Q-iteration

The aforesaid $Q$-functions $Q^b$ and $Q^*$ are characterized by the well-known Bellman equations as follows

$$Q^b(x, a) = \mathbb{E}\left\{ \rho(x, a, x') + \gamma Q^b(x', b(x')) \right\}, \quad (13)$$

$$Q^*(x, a) = \mathbb{E}\left\{ \rho(x, a, x') + \gamma \min_{a'} Q^*(x', a') \right\}. \quad (14)$$

The right-hand sides of each Bellman equation can be interpreted as mappings $T, T^b : \mathcal{Q} \to \mathcal{Q}$ applied to the corresponding $Q$-functions, where $\mathcal{Q}$ denotes the space of $Q$-functions. Thus, the Bellman equations (13) and (14) can be written as the following fixed-point relations

$$Q^b = T^b(Q^b), \quad Q^* = T(Q^*). \quad (15)$$
It is well known that $T$, as well as $T^k$, is a contraction with factor $\gamma < 1$ in $L_\infty$-norm [25, 28], that is,
\[
\|T(Q) - T(Q')\|_\infty \leq \gamma \|Q - Q'\|_\infty. \quad (16)
\]
Thus, $T$ has the unique fixed point $Q^*$ given by (15) [53, 54]. Accordingly, an arbitrary initial $Q$-function $Q_0$ can be iterated to eventually reach $Q^*$:
\[
Q_{k+1} = T(Q_k), \quad (17)
\]
which is known as the $Q$-iteration [23, 24, 28, 29]. Lastly, (15) and (16) yield
\[
\|Q_{k+1} - Q^*\|_\infty \leq \gamma \|Q_k - Q^*\|_\infty.
\]

### 4.3 Policy iteration convergence

Instead of iterating (17) at once, policy iteration first evaluates policies by constructing an estimate of state-action value function $Q^\phi$ of the current policy $b_k$ using (13), that is, it solves
\[
Q^\phi(x, a) = \mathbb{E}\{\rho(x, a, x') + \gamma Q^\phi(x', b_k(x'))\}. \quad (18)
\]
This $Q$-function is then used to construct a new improved policy via (9), which is greedy with respect to the state-action value function.

As seen from (18), policy iteration is linear in the $Q$-values, which is an important advantage of policy iteration over $Q$-iterations (17). For example, assuming the policy evaluation is exact, each policy improvement is guaranteed to find a strictly better policy unless it is already optimal [20, Chapter 11]. When finite state-action spaces are considered, the algorithm converges in a finite number of iterations since there is a finite number of possible policies.

Alas, policy evaluation typically cannot be solved exactly in continuous state-action spaces, and the $Q$-functions have to be approximated. The approximation (6) utilized herein is linear in parameters $\alpha_k$, which proves useful when solving the projected Bellman equation (10) and retains linearity. Following [28, Chapter 3], approximated policy iteration converts (17) into (cf. (10))
\[
\alpha_{k+1} = (P \circ T \circ F)(\alpha_k), \quad (19)
\]
where $F(\alpha)$ equals the right-hand side of (6) while the projection $P(Q)$ equals $\Phi(Q$ when orthonormal bases (e.g. Chebyshev polynomials) are employed. Let us analyze the expansiveness coefficients of $F$ and $P$. Using the induced operator norm of $\Phi^T(x, u)$, we reach
\[
\|F(\alpha) - F(\alpha')\|_\infty \leq \|\Phi^T(x, u)\|_\infty \|\alpha - \alpha'\|_\infty
\]

where $M$ and $N$ are the numbers of basis functions, that is, of Chebyshev polynomials, for approximating $x$ and $u$, respectively. Furthermore, upper bounding integrals for obtaining Chebyshev coefficients in each component of $P(Q) - P(Q^*)$ renders
\[
\|P(Q) - P(Q^*)\|_\infty \leq \pi \frac{L}{\gamma} \|Q - Q^*\|_\infty,
\]
making the expansiveness coefficient of $P \circ T \circ F$ in (19) upper bounded by
\[
E := \gamma \sqrt{2 \mu N M}. \quad (20)
\]
Clearly, the upper bound $E$ can be kept below unity by lowering $\gamma$ in Problem 1 as the approximation capabilities increase, which leads us to the following sufficient condition for algorithm convergence:

**Theorem 1.** If $E < 1$, then the composite mapping $P \circ T \circ F$ built upon Chebyshev polynomials is a contraction, that is, (19) converges to a unique fixed point.

The same procedure is applicable to general basis functions, even though orthonormal bases are more convenient leading to straightforward and elegant expressions for $E$ as illustrated herein. Moreover, this convergence result is somewhat unrealistic as it tacitly assumes ideal conditions (e.g. availability of an infinite number of data samples gathered with an adequate exploration), which motivates the following subsection. Also, notice that Theorem 1 supports the well-known guidelines regarding unwanted overfitting capabilities of the approximator [23, 24, 28]. Theorem 1 corresponds to the contribution (a) stated in the introduction.

### 4.4 Near-optimality bounds

Even though (18) can be solved iteratively (similarly to (17) since $T^k$ is a contraction), LSPI solves (18) at once by constructing the projected Bellman equation and employing the fact that it boils down to a system of linear equations (10). All details of the steps behind the transition from (6) and (18) to (10) are found in [32, 28, Chapter 3] and [24, Section 6.3]. Using the law-of-large-numbers arguments, [32] and [24, Section 6.3] show that (10) asymptotically converges to (18) with $Q$-functions projected to the space spanned by (6) as the number of samples $(x_i, \tau, r, x_{i+1})$ in (10)–(11) approaches infinity. Hence, as the experiments in Section 5 and simulations in [10] progress, the approximation $Q$ approaches $Q^*$ because more and more process data $(x_i, \tau, r, x_{i+1})$ are collected. Alas, not much can be said when merely a finite number of samples is available in (10), which is labelled the optimistic version of LSPI. This is the first of the two principal future research avenues on our agenda. However, as discussed in [24, Chapter 6], the optimistic version tends to explore better because with rapid changes of policy there is less tendency of bias towards particular states that are
favoured by a single policy. This subsection corresponds to the contribution \(\text{(b)}\) stated in the introduction.

Provided that we have collected enough samples, the second research avenue is related to the order of Chebyshev polynomials, that is, their approximation accuracy \([55]\). Namely, it is well known \([24]\) that if the approximate policy evaluation is accurate to within \(\delta\) in the \(L_\infty\) sense, that is, if

\[
\| \hat{Q}^{h_k} - Q^{h_k} \|_{\infty} \leq \delta, \quad \forall k \in \{1, 2, \ldots\}, \tag{20}
\]

then in the limit as \(k \to \infty\) the following near-optimality bound holds

\[
\limsup_{k \to \infty} \| \hat{Q}^{h_k} - Q' \|_{\infty} \leq \frac{2\gamma \delta}{(1 - \gamma)^2}. \tag{21}
\]

Moreover, if the sequence of obtained policies converges to some \(\bar{h}\), then the following tighter bound holds:

\[
\| \hat{Q}^{\bar{h}} - Q^* \|_{\infty} \leq \frac{2\gamma \delta}{1 - \gamma}. \tag{22}
\]

In other words, our second research avenue is to a priori select the order of Chebyshev polynomials that are \(\delta\)-accurate, in the sense of \((20)\), for the sequence of unknown functions \(Q^{h_k}, k \in \{1, 2, \ldots\}\). Clearly, since this is an online algorithm, computation-convergence trade-offs are inevitable. Fortunately, Section 3.1 ensures closed-loop stability along the way whilst Chebyshev approximators ensure fast convergence \([56]\).

5 | EXPERIMENTAL VERIFICATION

This section corresponds to the contribution \(\text{(c)}\) stated in Section 1.

5.1 | Flying area and positioning system

Experimental area of the Laboratory for Intelligent Autonomous Systems (LARIAT) is a \(4 \times 3 \times 2\) meter cuboid whose conceptual model is depicted in Figure 2. For localization, two HTC Vive Lighthouse stations \([57]\) and Bitcraze Lighthouse positioning decks mounted on each quadrotor (refer to Figure 3) are used. Bitcraze Lighthouse positioning decks are introduced by Bitcraze AB \([58]\) and, at the moment, they are still running in an early access program. Together with HTC Vive Lighthouse stations, they proved to be an affordable high-precision alternative to costly motion capture systems with positioning errors within 5 cm for our experimental setup. Both lighthouse stations sweep the space with laser lights in the horizontal and vertical directions at 120 Hz alternating fashion. To determine the relative angle of the positioning deck with respect to the stations, synchronization pulse is emitted before every sweep. When the positioning deck is reached by horizontal and vertical sweeps of both stations, the quadrotor position is calculated onboard as the intersection of two vectors pointing to the stations. Position measurements are supplied to the onboard extended Kalman filter (EKF) of each quadrotor at 30 Hz.

5.2 | Hardware

Flying vehicles used in the experiments are Crazyflie 2.1 nano quadrotors shown in Figure 3. These are open-source hardware and software development platforms manufactured by Bitcraze AB. With the 27 g take-off weight and dimension of 92 mm in diagonal, they are well suited for indoor MAS experiments. Each Crazyflie is equipped with an inertial measurement unit (IMU) for the three axis: accelerometer, gyroscope and magnetometer, together with a high-precision pressure sensor. This hardware was further extended with additional sensors, the aforementioned Lighthouse positioning decks, using the Crazyflie expansion port. Sensor readings and onboard computations are processed in a 32-bit, 168 MHz ARM microcontroller unit. The
Crazyradio PA is used on the base station PC side to establish a 2.4 GHz communication radio link to the quadrotor. Additional information about the Crazyflie 2.1 hardware design can be found in [58].

5.3 Firmware

The central part of the Crazyflie firmware is the stabilization loop. It coordinates the state estimation and control algorithms at 1000 Hz. State estimation is provided by an onboard running EKF [59, 60]. This EKF obtains the IMU measurements at 500 Hz and the aforementioned position measurements from the Bitcraze Lighthouse positioning deck at 30 Hz. Quadrotor position, velocity and attitude estimates are updated at 100 Hz. For position and attitude control, we use the proprietary PID controllers. For more regarding the Crazyflie firmware, the reader is referred to [61].

5.4 Base station

The relevant hardware specifications of our base station PC are: Intel® Xeon® Silver 4110 @ 2.1 GHz×16 processor and 32GB of RAM memory. The base station is running ROS Kinetic on Ubuntu 16.04. ROS packets used in the experiments are a part of the Crazyswarm project [62]. While ROS is responsible for handling communication between the quadrotors and base station, the LSPI algorithms are carried out in Mathworks Matlab. Matlab and ROS interface are established using Matlab Robotics System Toolbox as shown in Figure 4.

5.5 System identification

Identification of agents’ dynamics was carried out in two steps. In the first step, the velocity reference from Figure 5 was sent to each agent and positions of all agents were recorded. Subsequently, transfer functions in both x- and y-axis directions were obtained for all agents using the Matlab System Identification Toolbox. The parameters of transfer functions were estimated through iterative numerical search minimizing a weighted sum of squares of the errors between the model output and measured responses [63]. Since there are no major differences among the obtained transfer functions nor axes, we settle for the transfer function

\[ G(s) = \frac{K_p}{s(1 + T_p \cdot s)} \exp(-T_d \cdot s), \]

with \( K_p = 0.95312, T_p = 1.1266 \cdot 10^{-5} \) and \( T_d = 0.45 \) s for all agents in both x- and y-axis directions. Clearly, \( T_p \) is negligible so we set it to zero in what follows. It is to be pointed out that higher-order transfer functions did not show significant improvements in data fitting, that is, the improvements were below 0.5%.

The above transfer function was validated in the second step. Namely, every agent was given velocity reference individually and its position was recorded. The gathered data in x-axis direction are shown in Figure 5. A minimal mismatch between the model and measurements can be noticed owning primarily to initial positions not being exactly at zero. From the experiments, we conclude that the obtained transfer function captures the system dynamics quite well and can be used for model (1).

5.6 Consensus control

Altogether, we have a team of five agents with dynamics (cf. (1))

\[ \dot{\xi}_i = K_p r_i + \omega_i, \]

\[ \zeta_i = \xi_i, \]

and the communication topology shown in Figure 1, where the communication delay is \( d = 0.45 \) s. The corresponding graph
Laplacian matrix is

\[
L = \begin{bmatrix}
0 & 0 & 0 & 0 \\
-1 & 1 & 0 & 0 \\
-1 & 0 & 2 & 0 \\
0 & -1 & -1 & 2
\end{bmatrix}.
\]

Clearly, the first agent is the leader. In (2), we select \( K_1 = \cdots = K_9 = 0.35 \).

Using Section 3.1, the resulting \( \bar{\tau} \) for ZOH sampling is 5 ms. However, one cycle of the main Matlab script that implements Section 3.2 in \( \tau \)-axis and sampled-data periodic consensus in \( \gamma \)-axis, logs all experimental data and utilizes four Crazyradio PAs for communication among the base station and agents takes 6.46 ms to execute. In addition, the four Crazyradio PAs lead to worst-case communication period of 15 ms for a single agent, which corresponds to \( \bar{\tau} \) in our experiments and is the user-selected data exchange period in \( \gamma \)-axis. It is to be noted that the \( \tau \)-axis data is exchanged according to our LSPI at the upcoming multiple of 6.46 ms, which is yet another reason for a model-free approach since actions \( \tau \), effectively experience jitter (because 15 ms is not a multiple of 6.46 ms).

Clearly, \( \bar{\tau} \) of 5 ms with ZOH is not feasible (since it is smaller than 15 ms). One can mitigate this issue by investing in more advanced hardware (in order to decrease 6.46 and 15 ms), which is typically a costly and time-consuming option, or by making better use of the hardware at hand, which is the direction pursued herein. Hence, in an effort to obtain MATIs greater than \( \bar{\tau} = 15 \) ms, we introduce the following model-based estimator

\[
\hat{x}(t) = C^{cld} A^{cld} \hat{x}(t-\bar{d}),
\]  

(24)

which results in \( \bar{\tau} = 45 \) ms. Apparently, the delay \( \bar{d} \) is greater than the broadcasting interval \( \bar{\tau} \) (i.e. the so-called large delay). The impact of the values of \( K_i \)'s on \( \bar{\tau} \) in the delay-free setting is discussed in [22]. It is worth pointing out that [44] yields the MATI of 48 ms whereas the experimentally obtained sampled-data MATI is 50 ms (i.e. all agents exchange information at the same time every 50 ms). Clearly, the approach adopted herein for reaching \( \bar{\tau} \) is quite accurate.

The price paid for obtaining this relatively large \( \bar{\tau} \) of 45 ms is seen in (24). Namely, straightforward implementations of the model-based estimator (24) are in friction with the decentralized nature of our framework. Essentially, since \( C^{cld} A^{cld} \) is not block diagonal, (24) requires each agent to estimate the state of the entire group (not merely of itself and its neighbours as stated in the previous sections). To circumvent this nuisance in our experiments, the agents also exchange their local state estimates and forward state estimates of their neighbours (in addition to exchanging their own inputs and outputs as discussed in earlier sections). Surely, a more thought-out and formal approach is in order for larger groups and more complex topologies, but that is out of scope of the present work. For completeness, let us say that the greatest MATI obtained via purely decentralized model-based estimator, obtained after setting all non-diagonal entries of \( C^{cld} A^{cld} \) in (24) to zero, is merely 9 ms, which is an improvement upon 5 ms but still smaller than the aforementioned hardware limitation of 15 ms.

From [64, Corollary 3.4.], we infer that \( \tau_j^{|} \), \( j \in \{1, \ldots, N\} \), belongs to the interval \([-30, 30]\), which in turn defines \( \mathcal{X} \). The cost function parameters are \( y_1 = \cdots = y_N = 0.9 \), \( P_2 = 5 \), \( P_3 = P_4 = P_5 = 5N \), \( R_1 = \cdots = R_5 = 5 \), \( S_1 = S_2 = 50 \) and \( S_3 = 0 \). Notice that, since the first agent is the leader, its local discrepancy vector \( x_1 \) is not defined; hence, \( P_1 \) does not exist. The broadcasting intervals are initialized to \( \bar{\tau} \). Figure 6 provides the obtained numerical results suggesting that Theorem 1 is rather conservative in the sense of upper bounds on \( \gamma \) resulting in algorithm convergence.

A remark is in order regarding the choice of \( P_j \), \( R_j \) and \( S_j \). On the one hand, as we decrease \( S_j \) and keep \( P_j \) and \( R_j \) fixed, the obtained broadcasting intervals \( \tau_j^{|} \) approach \( \bar{\tau} \). On the other hand, as \( S_j \) becomes greater, \( \tau_j^{|} \) approach \( \bar{\tau} \) as seen from Figure 7a. Notice that this remark is expected and underlines the principal motivation for this work—overly sparse information exchange is not preferable a priori and needs to be applied thoughtfully, especially when the relative cost of communication with respect to the cost of control performance (e.g. stabilization or tracking performance) and control effort (e.g. movement) is low. However, the related literature does not take into account this cost viewpoint, but solely focuses on minimizing the communication effort.

A major source of disturbances in our experiments are nonlinear agents performing motions in close vicinity to each other. Namely, the downwash of each drone is quite strong and can destabilize other drones flying nearby of below it. Similarly, coupling effects among \( \tau \)- and \( \gamma \)-axis of each nano quadrotor are also disturbances.

Each agent learns its own approximation parameter vector \( \alpha_j^{|} \), which in turn renders the corresponding policy \( b_j^{|}(x_j(t_j^{|})) \) and actions \( t_j^{|} \)'s. Convergence of \( \alpha_j^{|} \)'s indicates a successful completion of the learning process (see Figure 7b), while \( t_j^{|} \)'s do not necessarily display smooth behaviors (see Figure 7a).

Furthermore, observe that \( \alpha_j^{|} \)'s are different for each agent even though homogeneous agents are encountered (which is the motivation behind utilizing homogeneous agents in this numerical example). Figure 8 shows the enclosed experiment
FIGURE 6 Experimental data obtained using the proposed methodology: (a) agents’ positions; (b) agents’ velocities and (c) velocity references along with velocities from (b) and the full video is available at [65]. Finally, note that somewhat periodic broadcasting instants stem from the stationary distributions and, to a certain extent, justify the traditional and widespread periodic feedback.

FIGURE 7 The process of learning: (a) suboptimal broadcasting intervals \( \tau_i \)'s; and (b) corresponding approximation parameter vectors \( \alpha_k \)'s for each agent updated every \( \kappa \) iterations.

FIGURE 8 Experimental verification snapshot

6 | CONCLUSION

This article studies MAS optimal intermittent feedback problems with LQR-like goal functions that capture local MAS performance versus agent lifetime trade-offs. Starting off with provably stabilizing upper bounds on agents’ broadcasting intervals, we bring together KF and an online off-policy model-free LSPI method to tackle coupled partially observable DP
problems. Subsequently, convergence and suboptimality of the proposed approach are addressed. Our framework is applicable to general heterogeneous linear agents with output feedback in the presence of disturbances, to realistic communication environments (e.g. intermittency, noise, delays and dropouts) as well as to directed and unbalanced communication topologies. Lastly, an off-the-shelf testbed is put together to demonstrate the benefits of proposed methodology.

In the future, our main goals are to investigate the impacts that finite number of samples and Chebyshev polynomials have on LSPI convergence and suboptimality. In this light, exploration-enhanced LSTD(Q(λ))-based PI algorithms will be considered owing to better understood convergence and noise robustness [24, Section 6.6]. In addition, it is of interest to tackle time-varying topologies and non-linear agent dynamics.

ACKNOWLEDGEMENTS

This work was supported in part by Croatian Science Foundation under the project IP-2016-06-2468 (ConDyS) and in part by the European Regional Development Fund under the grant KK.01.1.1.01.0009 (DATACROSS).

ORCID

Ivana Palunko https://orcid.org/0000-0001-5204-2294

REFERENCES

1. Olfati-Saber, R., Murray, R.M.: Consensus problems in networks of agents with switching topology and time-delays. IEEE Trans. Autom. Control 49(9), 1520–1533 (2004)
2. Ren, W., Beard, R.W.: Distributed consensus in multi-vehicle cooperative control - theory and applications. In: Communications and control engineering. Springer, London (2008)
3. Fang, L., Antsaklis, P.J.: On communication requirements for multi-agent consensus seeking. In: Antsaklis, P., Tabuada, P., (eds.) Networked Embedded Sensing and Control, Workshop NESC'05: University of Notre Dame, USA, 2005. vol. 331 of Lecture Notes in Control and Information Sciences. pp. 133–147. Springer (2006)
4. Xiao, F., Wang, L.: Asynchronous consensus in continuous-time multi-agent systems with switching topology and time-varying delays. IEEE Trans. Autom. Control 53(8), 1804–1816 (2008)
5. Yu, H., Antsaklis, P.J.: Output synchronization of multi-agent systems with event-driven communication: Communication delay and signal quantization. (2011). Technical Report of the ISIS Group at the University of Notre Dame, ISIS-2011-001, July 2011
6. Liu, Y., Jia, Y.: $H_\infty$ consensus control for multi-agent systems with linear coupling dynamics and communication delays. Int. J. Syst. Sci. 43(1), 50–62 (2012)
7. Guinaldo, M., et al.: Distributed event-based control strategies for interconnected linear systems. IET Control Theory Appl. 7(6), 877–886 (2013)
8. Songtan, E.D.: Clocks and insensitivity to small measurement errors. Int. J. Robust Nonlinear Control 4, 537–557 (1999)
9. Kellett, C.M., Teel, A.R.: Weak converse Lyapunov theorems and control-Lyapunov functions. SIAM J. Control Optim. 42(6), 1934–1959 (2004)
10. Tolić, D., Palunko, I.: Learning suboptimal broadcasting intervals in multi-agent systems. 20th IFAC World Congress, IFAC-PapersOnLine 50(1), 4144–4149 (2017)
11. Tolić, D., Hirsche, S.: Stabilizing transmission intervals for nonlinear delayed networked control systems. IEEE Trans. Autom. Control 62(1), 488–494 (2017)
12. Chen, T., Francis, B.A.: Optimal Sampled-Data Control Systems. Communications and Control Engineering, Springer-Verlag, Berlin, Heidelberg (1995) https://www.springer.com/gp/book/9781447130390
13. Xu, Y., Hespanha, J.P.: Optimal communication logics in networked control systems. In: IEEE Conf. on Decision and Control. (2004). pp. 3527–3532
14. Tolić, D., et al.: Optimal self-triggering for nonlinear systems via approximate dynamic programming. In: IEEE Multi-Conference on Systems and Control. (2012), pp. 879–884
15. Molin, A., Hirche, S.: On the optimality of certainty equivalence for event-triggered control systems. IEEE Trans. Autom. Control 58(2), 470–474 (2013)
16. Bini, E., Buttazzo, G.: The optimal sampling pattern for linear control systems. IEEE Trans. Autom. Control 59(1), 78–90 (2014)
17. Brunner, F.D., et al.: Stochastic thresholds in event-triggered control: A consistent policy for quadratic control. Automatica 89, 376–381 (2018)
18. I Balaghi, M.H., et al.: An l2-consistent data transmission sequence for linear systems. In: IEEE Conf. on Decision and Control. (2019). pp. 2622–2627
19. Tolić, D., Hirche, S.: Networked Control Systems with Intermittent Feedback. CRC Press, Boca Raton (2017)
20. Lewis, F.L., et al.: Optimal control. 3rd ed. In: Communications and Control Engineering. John Wiley & Sons, Inc., Hoboken (2012)
21. Mamdani, M.H., et al.: Error-dependent data scheduling in resource-aware multi-loop networked control systems. Automatica 81, 209–216 (2017)
22. Tolić, D., et al.: Resource management in cooperative multi-agent networks through self-triggering. IET Control Theory Appl. 9(6), 915–928 (2015)
23. Bertsekas, D.P.: Dynamic Programming and Optimal Control, vol. I, 4th ed. 4 Athena Scientific, Belmont, USA (2017)
24. Bertsekas, D.P.: Dynamic Programming and Optimal Control, vol. II, 4th ed. 4 Athena Scientific, Belmont, USA (2018)
25. Powell, W.B.: Approximate Dynamic Programming: Solving the Curses of Dimensionality. Wiley Series in Probability and Statistics. John Wiley and Sons, Hoboken, NJ, USA (2007)
26. Bertsekas, D.P., Tsitsiklis, J.N.: Neuro-Dynamic Programming. Athena Scientific, Belmont, USA (1996)
27. Sutton, R., Barto, A.: Reinforcement Learning: The MIT Press, Cambridge, USA (1998)
28. Buşoniu, L., et al.: Reinforcement Learning and Dynamic Programming Using Function Approximators. Automation and Control Engineering Series. CRC Press, Boca Raton, USA (2010)
29. Buşoniu, L., et al.: Reinforcement learning for control: Performance, stability, and deep approximators. Annu. Rev. Control 46, 8–28 (2018)
30. Palunko, I., et al.: A reinforcement learning approach for autonomous suspended load manipulation using aerial robots. In: IEEE Int. Conf. on Robotics and Automation. (2013). pp.4896–4901
31. Palunko, I., et al.: Cooperative suspended object manipulation using reinforcement learning and energy-based control. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems. (2014). pp. 885–891
32. Lagoudakis, M.G., Parr, R.: Least-squares policy iteration. J. Mach. Learn. Res. 4, 1107–1149 (2003)
33. Kamalapurkar, R., et al.: Approximate optimal cooperative decentralized control for consensus in a topological network of agents with uncertain nonlinear dynamics. In: IEEE American Control Conf. (2013). pp. 1320–1325
34. Alexander, H.L.: State estimation for distributed systems with sensing delay. In: SPIE Proceedings Vol. 1470, Data Structures and Target Classification. (1991). pp. 103–111
35. Kloz, J., et al.: Concurrent learning-based network synchronization. In: IEEE American Control Conf. (2014). pp. 796–801
36. Vamvoudakis, K.G., et al.: Multi-agent differential graphical games: Online adaptive learning solution for synchronization with optimality. Automatica 48, 1598–1611 (2012)
37. Vamvoudakis, K.G., et al.: Event-triggered optimal tracking control of nonlinear systems. Int. J. Robust Nonlinear Control 27(4), 598–619 (2017)
38. Tran, V.P., et al.: Distributed artificial neural networks-based adaptive strictly negative imaginary formation controller for unmanned aerial vehicles in time-varying environments. IEEE Trans. Ind. Inf. 1–1 (2020)
39. Li, J., et al.: Off-policy reinforcement learning for synchronization in multi-agent graphical games. IEEE Trans. Neural Networks Learn. Syst. 28(10), 2434–2445 (2017)
40. Hespanha, J.P., et al.: A survey of recent results in networked control systems. Proc. IEEE 95(1), 138–162 (2007)
41. Tolić, D., Fierro, R.: Adaptive sampling for tracking in pursuit-evasion games. In: IEEE Multi-Conference on Systems and Control. (2011). pp. 179–184
42. Gaitsgory, V., et al.: Stabilization with discounted optimal control. Syst. Control Lett. 82, 91–98 (2015)
43. Postoyan, R., et al.: Stability analysis of discrete–time infinite-horizon optimal control with discounted cost. IEEE Trans. Autom. Control 62(6), 2736–2749 (2017)
44. Tolić, D.: Stabilizing transmission intervals and delays in nonlinear networked control systems through hybrid-system-with-memory modeling and Lyapunov-Krasovskii arguments. Nonlinear Anal. Hybrid Syst 36, 100834 (2020)
45. Khasminskii, R.: Stochastic Stability of Differential Equations. Stochastic Modelling and Applied Probability. Stochastic Modelling and Applied Probability, 2 Springer, Berlin, Heidelberg (2012) https://www.springer.com/gp/book/9783642232794
46. Tolić, D., et al.: Decentralized cooperative control in degraded communication environments. In: Vamvoudakis, K.G., Jagannathan, S. (eds.). Control of Complex Systems: Theory and Applications, pp. 373–395.Elsevier, Oxford (2016)
47. Bellman, R.E.: Dynamic Programming. Princeton University Press, New Jersey (1957)
48. Poor, H.V.: An Introduction to Signal Detection and Estimation. 2nd ed. 2 Springer-Verlag, New York (1994) https://www.springer.com/gp/book/9780387941738
49. Khasminskii, R.: Stochastic Stability of Differential Equations. Stochastic Modelling and Applied Probability. Stochastic Modelling and Applied Probability, 2 Springer, Berlin, Heidelberg (2012) https://www.springer.com/gp/book/9783642232794
50. Preiss, J.A., et al.: Crazyswarm GitHub. (2019). https://github.com/USC-ACTLab/crazyswarm. Accessed 19 March 2019
51. Anderson, B.D.O., Moore, J.B.: Detectability and stabilizability of time-varying discrete-time linear systems. SIAM J. Control Optim. 19(1), 20–32 (1981)
52. Golub, G.H., Loan, C.F.V.: Matrix Computations, 3rd ed. 3 The John Hopkins University Press, Baltimore, Maryland, USA (1996)
53. Puterman, M.L.: Markov Decision Processes. Wiley, Hoboken (1994)
54. Istratescu, V: Fixed point theorem. Math. Appl. 7, (1998)
55. Heston, T., et al.: The Elements of Statistical Learning: Data Mining, Inference and Prediction, 2nd ed. Springer, Berlin (2009)
56. Boyd, J.P.: Chebyshev and Fourier Spectral Methods. Dover Publications, Mineola (2001)
57. Niehorster, D.C., Li, L., Lapte, M.: The accuracy and precision of position and orientation tracking in the HTC Vive virtual reality system for scientific research. Iperception 8(3), 1–23 (2017)
58. Bitcraze. Bitcraze Wiki. (2019). https://wiki.bitcraze.io/projects/crazyflie2: index. Accessed 20 May 2019
59. Mueller, M.W., et al.: Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadrocopter state estimation. In: 2015 IEEE International Conference on Robotics and Automation (ICRA). (2015). pp. 1730–1736
60. Mueller, M.W., et al.: Covariance correction step for kalman filtering with an attitude. J. Guid. Control Dyn. 40(9), 2301–2306 (2017)
61. Istratescu, V: Fixed point theorem. Math. Appl. 7, (1998)
55. Heston, T., et al.: The Elements of Statistical Learning: Data Mining, Inference and Prediction, 2nd ed. Springer, Berlin (2009)
56. Boyd, J.P.: Chebyshev and Fourier Spectral Methods. Dover Publications, Mineola (2001)
57. Niehorster, D.C., Li, L., Lapte, M.: The accuracy and precision of position and orientation tracking in the HTC Vive virtual reality system for scientific research. Iperception 8(3), 1–23 (2017)
58. Bitcraze. Bitcraze Wiki. (2019). https://wiki.bitcraze.io/projects/crazyflie2: index. Accessed 20 May 2019
59. Mueller, M.W., et al.: Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadrocopter state estimation. In: 2015 IEEE International Conference on Robotics and Automation (ICRA). (2015). pp. 1730–1736
60. Mueller, M.W., et al.: Covariance correction step for kalman filtering with an attitude. J. Guid. Control Dyn. 40(9), 2301–2306 (2017)
61. Istratescu, V: Fixed point theorem. Math. Appl. 7, (1998)
55. Heston, T., et al.: The Elements of Statistical Learning: Data Mining, Inference and Prediction, 2nd ed. Springer, Berlin (2009)
56. Boyd, J.P.: Chebyshev and Fourier Spectral Methods. Dover Publications, Mineola (2001)
57. Niehorster, D.C., Li, L., Lapte, M.: The accuracy and precision of position and orientation tracking in the HTC Vive virtual reality system for scientific research. Iperception 8(3), 1–23 (2017)
58. Bitcraze. Bitcraze Wiki. (2019). https://wiki.bitcraze.io/projects/crazyflie2: index. Accessed 20 May 2019
59. Mueller, M.W., et al.: Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadrocopter state estimation. In: 2015 IEEE International Conference on Robotics and Automation (ICRA). (2015). pp. 1730–1736
60. Mueller, M.W., et al.: Covariance correction step for kalman filtering with an attitude. J. Guid. Control Dyn. 40(9), 2301–2306 (2017)
61. Istratescu, V: Fixed point theorem. Math. Appl. 7, (1998)
55. Heston, T., et al.: The Elements of Statistical Learning: Data Mining, Inference and Prediction, 2nd ed. Springer, Berlin (2009)
56. Boyd, J.P.: Chebyshev and Fourier Spectral Methods. Dover Publications, Mineola (2001)
57. Niehorster, D.C., Li, L., Lapte, M.: The accuracy and precision of position and orientation tracking in the HTC Vive virtual reality system for scientific research. Iperception 8(3), 1–23 (2017)
58. Bitcraze. Bitcraze Wiki. (2019). https://wiki.bitcraze.io/projects/crazyflie2: index. Accessed 20 May 2019
59. Mueller, M.W., et al.: Fusing ultra-wideband range measurements with accelerometers and rate gyroscopes for quadrocopter state estimation. In: 2015 IEEE International Conference on Robotics and Automation (ICRA). (2015). pp. 1730–1736
60. Mueller, M.W., et al.: Covariance correction step for kalman filtering with an attitude. J. Guid. Control Dyn. 40(9), 2301–2306 (2017)
61. Istratescu, V: Fixed point theorem. Math. Appl. 7, (1998)