Shared Linear Quadratic Regulation Control: A Reinforcement Learning Approach

Murad Abu-Khalaf, Sertac Karaman, and Daniela Rus

Abstract— **We propose controller synthesis for state regulation problems in which a human operator shares control with an autonomy system, running in parallel. The autonomy system continuously improves over human action, with minimal intervention, and can take over full-control if necessary. It additively combines user input with an adaptive optimal corrective signal to drive the plant. It is adaptive in the sense that it neither estimates nor requires a model of the human's action policy, or the internal dynamics of the plant, and can adjust to changes in both. Our contribution is twofold; first, a new controller synthesis for shared control which we formulate as an adaptive optimal control problem for continuous-time linear systems and solve it online as a human-in-the-loop reinforcement learning. The result is an architecture that we call *shared* linear quadratic regulator (sLQR). Second, we provide new analysis of reinforcement learning for continuous-time linear systems in two parts. In the first analysis part, we avoid learning along a single state-space trajectory which we show leads to data collinearity under certain conditions. In doing so, we make a clear separation between exploitation of learned policies and exploration of the state-space, and propose an exploration scheme that requires switching to new state-space trajectories rather than injecting noise continuously while learning. This avoidance of continuous noise injection minimizes interference with human action, and avoids bias in the convergence to the stabilizing solution of the underlying algebraic Riccati equation. We show that exploring a minimum number of pairwise distinct state-space trajectories is necessary to avoid collinearity in the learning data. In the second analysis part, we show conditions under which existence and uniqueness can be established for off-policy reinforcement learning in continuous-time linear systems; namely a required knowledge of the input matrix.**

I. INTRODUCTION

We address technical challenges associated with adaptation and optimization that emerge when a human operator shares low-level regulation control tasks with an automatic control system. The rapid acceleration in machine learning and artificial intelligence at large creates opportunities to operate in a data-driven response-based manner, rather than a model-based one, and without a priori knowledge of the human-in-the-loop action policy. We wish to investigate these notions for a fundamental building block of control system design, *e.g.* the linear quadratic regulator (LQR). The theory is inspired by applications in which a human operator, assisted by an autonomy system, is regulating a steering angle, speed, or spacing between vehicles, among others, or balancing objects including the human body itself.

Murad Abu-Khalaf and Daniela Rus are with the MIT Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139 USA. {murad,rus}@csail.mit.edu

Sertac Karaman is with the Laboratory of Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139 USA. sertac@mit.edu

Different levels of shared autonomy are reported in the literature. In robotic teleoperation, shared autonomy augments a user's ability to control a robot via an interface that generally has less degrees of freedom than the robot itself. Many such paradigms of shared autonomy require the autonomy system to predict user intentions and augment user input via a policy arbitration scheme. The authors in [1] proposed policy-blending as a "common lens" to understand policy arbitration and control effort division across time or tasks while [2] proposes to handle a distribution of goals at once by formulating a Partially Observable Markov Decision Problem (POMDP). Modeling assumptions for the underlying POMDP and the user policy have been relaxed in [3] by leveraging a human-in-the-loop deep Q-learning.

In [4], [5], shared control for semi-autonomous driving is formulated as a Model Predictive Control (MPC) problem. Its cost function does not account for human input and uses instead a form of policy arbitration. In [6], an MPC problem to deal with obstacle avoidance is formulated that handles policy arbitration implicitly, and uses quadratic costs that consider human input. In [7], parallel autonomy for safe driving is formulated as an MPC problem that handles policy arbitration implicitly as well. It estimates the user input and holds it constant while evolving the dynamics over the prediction horizon. It accounts for the user input in the cost function and uses a forgetting factor to emphasize short-term impact of user input. All these MPC approaches are model-based, and human input when considered is provided as an estimate that is held constant over the prediction horizon.

In [8], aircraft control tasks are shared between a human pilot and an adaptive autopilot. The pilot assumes high-level tasks, detection of anomalies and switching of controller structure, and relegates low-level regulation to the autopilot.

In this paper, a human operator shares low-level regulation control task with an adaptive parallel autonomy system via a human-computer interface (HCI), and without requiring a model of the human policy or the plant's internal dynamics. We formulate the problem in an optimal control theoretic sense, and solve the underlying dynamic programming problem online via reinforcement learning. The objective of the autonomy system is to assist the human operator by improving closed-loop performance without significantly deviating from user input, and to take over from the user when necessary. It serves as a secondary controller or co-pilot, that optimally modulates with an additive corrective signal the output of a primary controller enacted by a human operator; therefore, policy arbitration here is implicit. We refer to the resulting architecture as a shared linear quadratic regulator (sLQR). The sLQR emerges as a solution to the underlying algebraic Riccati equation (ARE) for the human-in-the-loop closed-loop dynamics.

Our approach is not model-based but one that is driven by the system response. We dissect the role of exploration from exploitation to eliminate sources of collinearity in the learning data, and the need to continuously inject perturbation noise as common in the literature. Our focus is on continuous-time linear systems. A quadratic form is used to capture the Lyapunov function or cost-to-go of the optimization problem resulting in essentially learning algorithms that involve polynomial regression. For more general nonlinear dynamics with state and input constraints, extensions are possible via the use of neural networks and approximate dynamic programming tools from [11].

In Section I-A, we introduce the notation used throughout the paper. In Section II, we formulate shared control as an adaptive optimal control problem for state regulation. In Section III we review relevant background on reinforcement learning useful to our shared control methodology and point out existing gaps in the literature. In Section IV we address some existing gaps by providing new solvability analysis for policy iterations. In Section V, we solve the formulated shared control problem and present the sLQR. In Section VI we apply sLQR to a car-following application. In Section VII, we provide some conclusions and future directions.

A. Notation

$\mathbb{R}$ denotes the real line and $\mathbb{C}$ the set of complex numbers. Given matrix $Y \in \mathbb{R}^{p\times q}$, $Y_{([1:p],j)}$ denotes its j^{th} column and $Y_{([i_1:i_2],j)}$ denotes a column vector formed from the j^{th} column of Y starting at row i_1 and ending at row i_2. The element at the i^{th} row and j^{th} column is denoted by $Y_{(i,j)}$. From [19], $vec(\cdot)$ stacks the columns of Y from first to last into a single column of size $pq \times 1$ as shown, and the operator $\otimes$ is the Kronecker product where given matrix Z of an arbitrary size

$$vec(Y) \triangleq \begin{bmatrix} Y_{([1:p],1)} \\ Y_{([1:p],2)} \\ \vdots \\ Y_{([1:p],q)} \end{bmatrix}, Y\otimes Z \triangleq \begin{bmatrix} Y_{(1,1)}Z & \dots & Y_{(1,q)}Z \\ \vdots & \ddots & \vdots \\ Y_{(p,1)}Z & \dots & Y_{(p,q)}Z \end{bmatrix}$$

We define $vec^L(\cdot)$ which operates on a square matrix of size $n \times n$ by stacking from first to last the columns of its lower triangular part into a single column of size $\frac{n(n+1)}{2} \times 1$ as shown for $Z \in \mathbb{R}^{n\times n}$

$$vec^L(Z) \triangleq \begin{bmatrix} Z_{([1:n],1)} \\ Z_{([2:n],2)} \\ \vdots \\ Z_{([n:n],n)} \end{bmatrix}.$$

Moreover, $\kappa(i,j)$ denotes the index of $Z_{(i,j)} \in vec^L(Z)$. For example, $\kappa(1,1) = 1$ and $\kappa(n,n) = \frac{n(n+1)}{2}$.

If $x \in \mathbb{R}^{n\times 1}$, then $x \otimes x = vec(xx^\intercal)$ where $x^\intercal$ is the transpose of x. If additionally $P \in \mathbb{R}^{n\times n}$, then $x^\intercal P x = (x\otimes x)^\intercal vec(P) = vec(xx^\intercal)^\intercal vec(P) = vec^L(xx^\intercal)^\intercal W$, where $W \in \mathbb{R}^{\frac{n(n+1)}{2}\times 1}$. For all $w_{\kappa(i,j)} \in W$, $w_{\kappa(i,j)} = p_{ij} + p_{ji} - \delta_{ij}p_{ij}$ where δ_{ij} is the Kronecker delta function. This follows from matching the terms of

$$vec^L(xx^\intercal)^\intercal W = \sum_{j=1}^{n}\sum_{i=j}^{n} w_{I(i,j)}x_i x_j,$$

$$vec(xx^\intercal)^\intercal vec(P) = \sum_{j=1}^{n}\sum_{i=j}^{n} (p_{ij} + p_{ji} - \delta_{ij}p_{ij})\, x_i x_j.$$

An all zero-entries $n \times m$ matrix is denoted by $\mathbf{0}_{n\times m}$, and by $\mathbf{0}$ when the size is context-dependent. Similarly, $\mathbf{I}_n$ denotes a size n identity matrix, and $\mathbf{I}$ is used when the size is context-dependent. Moreover, we denote by $\mathbf{e}_{ij} \in \mathbb{R}^{m\times n}$ a matrix with 1 at the i^{th} row and j^{th} column and 0 otherwise.

II. Problem Formulation

Consider a continuous-time linear system given by

$$\dot{x}(t) = Ax(t) + Bu(t), \tag{1a}$$
$$y_a = x, \tag{1b}$$
$$y_h = C_h x, \tag{1c}$$
$$u = u_h + u_a, \tag{1d}$$

with $x \in \mathbb{R}^{n\times 1}$ and $u \in \mathbb{R}^{m\times 1}$; $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ and $C_h \in \mathbb{R}^{p\times n}$ are constant matrices with (A, B) stabilizable, and (A, B, C_h) static output feedback stabilizable. $y_a \in \mathbb{R}^{n\times 1}$ is the plant output available to the autonomy system, while $y_h \in \mathbb{R}^{p\times 1}$ is the plant output available to the human operator. The input to the plant u is provided by a human-computer interface and is decomposed into a human-generated u_h and an autonomy-computed u_a per (1d).

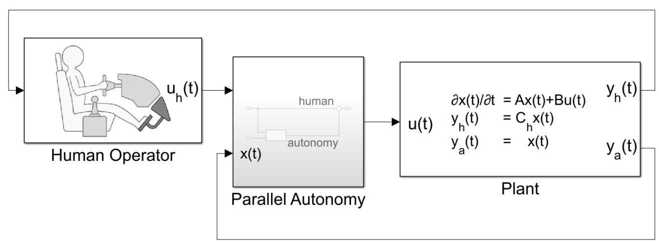

Fig. 1. Parallel Autonomy.

The human's action policy is unknown to the autonomy system which can observe and measure u_h. It is assumed that the human operator observes the plant via $y_h = C_h x$ and that the human implements a linear static output feedback control to stabilize the plant as follows $u_h(x) = K_h y_h(x) = K_h C_h x$. Thus, the human policy is linear in the state for an observation matrix C_h and gain K_h that are unkown to the autonomy system. Unlike the human operator, the autonomy system has full access to the state of the plant, *i.e.* $y_a = x$.

The aim of the autonomy system is to compute and apply u_a to meet two objectives: First, to improve the human-in-the-loop closed-loop performance with minimum intervention as experienced by the user. Second, to entirely take over control from the human operator when necessary. These two objectives are stated concisely as Problem 1 and Problem 2.

Problem 1. *Consider the parallel autonomy system* (1). *Solve the infinite-horizon optimal control problem*

$$J(x_0, t_0, u_h) = \inf_{u_a} \int_{t_0}^{\infty} (x^\intercal Q x + u_h^\intercal(x) M u_h(x) + u_a^\intercal R u_a)\, dt, \tag{2}$$

where $x_0 = x(0)$, without the knowledge of the plant's internal dynamics matrix A, the human's observation matrix C_h and the human output feedback gain K_h, and adapting to subsequent changes in A, K_h, and C_h. The optimization is with respect to u_a, and is not *with respect to u.*

Problem 2. *To compute an optimal takeover control u_a for (1a) such that u_a solves (2) under the additional condition that $u_h = 0$. In this case, there is* no *human-in-the-loop.*

III. Preliminaries on Policy Iterations

In this section, we highlight results and gaps in existing policy iterations literature useful to our proposed shared control approach. We later address the gaps in Section IV.

A. Offline Model-Based Policy Iterations

This type of policy iterations is executed offline without generating a system response. It is model-based requiring full-knowledge of the system matrices A and B. In this case, given $u_i(t) = K_i x(t)$ with stabilizing $K_i \in \mathbb{R}^{m\times n}$, one has state-space trajectories $\varphi(t, x_0, u_i(t))$ where $x_0 = x(0)$ and such that

$$x(t) = e^{(A+BK_i)(t-t_0)} x(t_0). \tag{3}$$

A quadratic cost-to-go for the policy u_i is obtained by integrating over $\varphi(t, x_0, u_i(t))$ as follows

$$\begin{aligned} J_i(x_0, t_0) &= \int_{t_0}^{\infty} (x^\intercal(t) Q x(t) + u_i^\intercal(t) R u_i(t))\, dt \\ &= x(t_0)^\intercal P_i x(t_0) = V_i(x(t_0)), \end{aligned} \tag{4}$$

where

$$P_i = \int_{t_0}^{\infty} \left[e^{(A+BK_i)(t-t_0)}\right]^\intercal [Q + K_i^\intercal R K_i]\, e^{(A+BK_i)(t-t_0)} dt. \tag{5}$$

It follows from (5) that P_i satisfies

$$\begin{aligned} &P_i(A + BK_i) + (A + BK_i)^\intercal P_i = \\ &\int_{t_0}^{\infty} \frac{d}{dt}\left(\left[e^{(A+BK_i)(t-t_0)}\right]^\intercal [Q + K_i^\intercal R K_i]\, e^{(A+BK_i)(t-t_0)}\right) dt \\ &= -\left[Q + K_i^\intercal R K_i\right]. \end{aligned} \tag{6}$$

and therefore P_i satisfies a Lyapunov matrix equation and thus can be solved for directly without requiring integration over the state-space trajectories under the assumption that the Lyapunov matrix equation has a unique solution. Moreover, $V_i(x)$ serves as a Lyapunov function for $u_i(x)$ satisfying (7a)

$$\frac{dV_i}{dx}^\intercal (Ax + Bu_i) = -x^\intercal Q x - u_i^\intercal R u_i, \tag{7a}$$

$$u_{i+1} = -\tfrac{1}{2} R^{-1} B^\intercal \frac{dV_i}{dx}, \tag{7b}$$

where (7b) is the policy iteration update that alos uses $V_i(x)$.

It was shown in [9] that following proper initialization, iterating on (8a) and (8b) converges to the stabilizing solution of the algebraic Riccati equation (8c) denoted by $\mathcal{P}$

$$P_i(A + BK_i) + (A + BK_i)^\intercal P_i + K_i^\intercal R K_i + Q = 0, \tag{8a}$$

$$K_{i+1} = -R^{-1} B^\intercal P_i, \tag{8b}$$

$$PA + A^\intercal P - P^\intercal B^\intercal R^{-1} B^\intercal P + Q = 0, \tag{8c}$$

where $\mathcal{P}$ is the quadratic value function matrix

$$\begin{aligned} V(x(t_0)) &= \inf_u \int_{t_0}^{\infty} (x^\intercal(t) Q x(t) + u^\intercal R u)\, dt \\ &= x(t_0)^\intercal \mathcal{P} x(t_0). \end{aligned}$$

B. Response-Based Data-Driven Policy Iterations

1) On-Policy: In [12], the right-hand-side of (4) is split into two parts

$$\begin{aligned} V_i(x(t_k)) = &\int_{t_k}^{t_k+\tau} (x^\intercal(t) Q x(t) + u_i^\intercal(t) R u_i(t))\, dt \\ &+ V_i(x(t_k + \tau)). \end{aligned} \tag{9}$$

Substitute $V_i(x) = x^\intercal P_i x$ in (9) to get

$$\begin{aligned} &x(t_k + \tau)^\intercal P_i x(t_k + \tau) - x(t_k)^\intercal P_i x(t_k) \\ &= -\int_{t_k}^{t_k+\tau} (x^\intercal(t) Q x(t) + u_i^\intercal(t) R u_i(t))\, dt. \end{aligned} \tag{10}$$

The result is a response-based data-driven approach to solve for the unknown P_i given online measurements. This online data, indexed by k, has two parts; the first part is state measurements $x(t_k)$ and $x(t_k + \tau)$ substituted on the left-hand side of (10); the second is the reinforcement signal experienced over a finite horizon $[t_k, t_k + \tau]$ as represented by the integral on the right-hand side of (10). This results in a linear system of unknowns, namely the coefficients of a quadratic form, that can be solved for by polynomial regression. Unlike (8a), equation (10) does not require A and B of (1a) to solve for P_i. Knowledge of B is still required to perform the policy update (8b).

In [12], the linear system of unknowns (10) is written element-wise using Kronecker products such that the unknowns, *i.e.* the elements of P_i, form a vector. For the purpose of polynomial regression, dependent regressors should be eliminated, *i.e.* $x_i x_j$ and $x_j x_i$ are dependent. The lower triangular part of $xx^\intercal$ represents all independent regressors. In this regard, and using the notation of Section I-A, we use a quadratic form $V(x) = \mathbf{w}^\intercal \phi(x)$, where $\phi(x) = vec^L(xx^\intercal)$, to solve for $\mathbf{w}$ from measurements by forming the following linear system of unknowns $\mathbf{A}\mathbf{w} = \mathbf{b}$. The measurement matrices $\mathbf{A} \in \mathbb{R}^{N\times\frac{n(n+1)}{2}}$ and $\mathbf{b} \in \mathbb{R}^{N\times 1}$ are

$$\mathbf{A} = \begin{bmatrix} \phi(x_\tau[0])^\intercal - \phi(x[0])^\intercal \\ \vdots \\ \phi(x_\tau[k])^\intercal - \phi(x[k])^\intercal \\ \vdots \\ \phi(x_\tau[N-1])^\intercal - \phi(x[N-1])^\intercal \end{bmatrix}, \tag{11a}$$

$$\mathbf{b} = -\left[\ r_0, \ldots, r_k, \ldots, r_{N-1}\ \right]^\intercal, \tag{11b}$$

$$r_k = \int_{t_{x[k]}}^{t_{x[k]}+\tau} (x^\intercal(t) Q x(t) + u_i^\intercal(t) R u_i(t))\, dt \tag{11c}$$

where the k^{th} state measurements pair $x_\tau[k]$ and $x[k]$ and the reward r_k are such that r_k is integrated over a state-space trajectory segment evolving per (3) such that $x_\tau[k] = e^{(A+BK_i)\tau} x[k]$. Note that unlike [12], we do not assume that $x[0], \ldots, x[k], \ldots, x[N-1]$ trace the same state-space trajectory $\varphi(t, x_0, u_i(t))$. Note also that if the dependent regressors are included, then the columns of $\mathbf{A}$ will be dependent. Moreover, $N \geq \frac{n(n+1)}{2}$ and the linear system of unknowns (11) is *consistent*, and $\mathbf{w} = \mathbf{A}^{-1}\mathbf{b}$.

Remark 1. *It was suggested in [12] – as well as in [10] where the special case of letting $\tau = \infty$ in (10) is treated – that data collection can be along a single state-space trajectory $\varphi(t, x_0, u_i(t))$. In that case, the k^{th} row of (11a) is $\phi(x(t_k+\tau))^\intercal - \phi(x(t_k))^\intercal$ and $r_k = \int_{t_k}^{t_k+\tau}(x^\intercal(t)Qx(t) + u_i^\intercal(t)Ru_i(t))\,dt$ for (11b). In Section IV, we show conditions under which collecting data along the same state-trajectory causes data collinearity; we suggest an adjusted controller synthesis to avoid data collinearity.*

2) Off-Policy: Given the cost-to-go $V_i(x)$ for policy $u_i(x)$ as determined by (4). Differentiating $V_i(x)$ over the trajectories $\varphi(t, x_0, u(t))$ generated by another policy $u(t) = Fx(t)$ such that

$$x(t) = e^{(A+BF)(t-t_0)}x(t_0), \tag{12}$$

we then get

$$\begin{aligned} \dot{V}_i &= \frac{dV_i}{dx}^\intercal(Ax + Bu) \\ &= \frac{dV_i}{dx}^\intercal(Ax + Bu_i) + \frac{dV_i}{dx}^\intercal B\Delta(u, u_i) \\ &= -x^\intercal Qx - u_i^\intercal Ru_i + \frac{dV_i}{dx}^\intercal B\Delta(u, u_i), \end{aligned} \tag{13}$$

where $\Delta(u, u_i) = u - u_i$. In [11], eq. (13) is used to show that V_i serves as a Lyapunov function to the dynamics driven by the improved policy $u_{i+1} = -\frac{1}{2}R^{-1}B^\intercal\frac{dV_i}{dx}$, *i.e.* (8b), namely

$$\begin{aligned} \dot{V}_i &= -x^\intercal Qx - u_i^\intercal Ru_i + \frac{dV_i}{dx}^\intercal B(u_{i+1} - u_i) \\ &= -x^\intercal Qx - u_i^\intercal Ru_i - (u_{i+1} - u_i)^\intercal R(u_{i+1} - u_i). \end{aligned}$$

In [13], eq. (13) is integrated over $\varphi(t, x_0, u(t))$ as follows

$$\int_{t_k}^{t_k+\tau}\dot{V}_i dt = \int_{t_k}^{t_k+\tau}\frac{dV_i}{dx}^\intercal(Ax + Bu)\,dt,$$

leading to

$$\begin{aligned} V_i(x(t_k+\tau)) - V_i(x(t_k)) - \int_{t_k}^{t_k+\tau}\frac{dV_i}{dx}^\intercal B\Delta(u, u_i)dt \\ = -\int_{t_k}^{t_k+\tau}(x^\intercal Qx + u_i^\intercal Ru_i)\,dt. \end{aligned} \tag{14}$$

Thus $V_i(x)$ satisfies both eqs. (9) and (14) where in the former the system is driven by $u_i(x)$, and in the latter it is driven by $u(x)$. Note that in this off-policy version, the closed-loop dynamics does not update and is fixed from one policy iteration to another.

Remark 2. *In [13], the policy update relation (7b) is embedded in (14) to eliminate the explicit knowledge of B requirement and to solve simultaneously for V_i and u_{i+1} directly as follows*

$$\begin{aligned} \hat{V}_i(x(t_k+\tau)) - \hat{V}_i(x(t_k)) + 2\int_{t_k}^{t_k+\tau}\Delta(u, u_i)^\intercal R\hat{u}_{i+1}dt \\ = -\int_{t_k}^{t_k+\tau}(x^\intercal Qx + u_i^\intercal Ru_i)\,dt. \end{aligned} \tag{15}$$

The reference [13] and subsequent work [14]–[16] assume that (15) has a unique solution pair $\{V_i, u_{i+1}\}$ corresponding to the solution sequence determined by (8a) and (8b). In Section IV, we show that while (14) has a unique solution that corresponds to the correct cost-to-go for V_i, (15) has nonunique solutions one of them corresponds to the cost-to-go V_i for u_i.

Substitute $V_i(x) = x^\intercal P_i x$ and $\Delta(u, u_i) = Fx - K_i x = L_i x$ in (14) to get

$$\begin{aligned} x(t_k+\tau)^\intercal P_i x(t_k+\tau) - x(t_k)^\intercal P_i x(t_k) \\ - \int_{t_k}^{t_k+\tau}2x(t)^\intercal P_i BL_i x(t)dt \\ = -\int_{t_k}^{t_k+\tau}(x^\intercal(t)Qx(t) + u_i^\intercal(t)Ru_i(t))\,dt. \end{aligned} \tag{16}$$

The result is a response-based data-driven approach similar to (10) to solve for the unknown P_i given online measurements.

Using the notation of Section I-A, we use a quadratic form $V(x) = \mathbf{w}^\intercal\phi(x)$, where $\phi(x) = vec^L(xx^\intercal)$, and $\frac{dV}{dx} = \nabla\phi(x)^\intercal\mathbf{w}$ to solve for W from measurements by forming the following linear system of unknowns $\mathbf{A}\mathbf{w} = \mathbf{b}$; where $\mathbf{A} \in \mathbb{R}^{N\times\frac{n(n+1)}{2}}$ and $\mathbf{b} \in \mathbb{R}^{N\times 1}$ are such that

$$\mathbf{A} = \begin{bmatrix} \phi(x_\tau[0])^\intercal - \phi(x[0])^\intercal - \delta_0 \\ \vdots \\ \phi(x_\tau[k])^\intercal - \phi(x[k])^\intercal - \delta_k \\ \vdots \\ \phi(x_\tau[N-1])^\intercal - \phi(x[N-1])^\intercal - \delta_{N-1} \end{bmatrix}, \tag{17a}$$

$$\mathbf{b} = -\begin{bmatrix} r_0, \ldots, r_k, \ldots, r_{N-1} \end{bmatrix}^\intercal, \tag{17b}$$

$$\delta_k = \int_{t_{x[k]}}^{t_{x[k]}+\tau}x(t)^\intercal L_i^\intercal B^\intercal\nabla\phi(x(t))^\intercal dt, \tag{17c}$$

$$r_k = \int_{t_{x[k]}}^{t_{x[k]}+\tau}(x^\intercal(t)Qx(t) + u_i^\intercal(t)Ru_i(t))\,dt, \tag{17d}$$

where the k^{th} state measurements pair $x_\tau[k]$ and $x[k]$ and the reward r_k are such that r_k is integrated over a state-space trajectory segment evolving per (12) such that $x_\tau[k] = e^{(A+BF)\tau}x[k]$. We do not assume that $x[0], \ldots, x[k], \ldots, x[N-1]$ trace the same state-space trajectory $\varphi(t, x_0, u(t))$. Moreover, $N \geq \frac{n(n+1)}{2}$ and the linear system of unknowns (17) is *consistent*.

In off-policy iterations, $x(t)$ evolves per $u = Fx$ and is therefore independent of u_i policy updates. Therefore, the closed-loop data collected at the initial iteration $i = 0$ can be used in subsequent iterations by recomputing K_i dependent terms in (17). To do so, one can re-write (17c) and (17d) by factoring out K_i using $K_i = \sum_{p=1}^n\sum_{q=1}^m[K_i]_{(p,q)}\mathbf{e}_{pq}$ such that,

$$\delta_k = \delta_k(F) - \delta_k(K_i), \tag{18a}$$

$$\delta_k(F) = \int_{t_{x[k]}}^{t_{x[k]}+\tau}u(x(t))^\intercal B^\intercal\nabla\phi(x(t))^\intercal dt, \tag{18b}$$

$$\delta_k(\mathbf{e}_{pq}) = \int_{t_{x[k]}}^{t_{x[k]}+\tau}x(t)^\intercal\mathbf{e}_{pq}^\intercal B^\intercal\nabla\phi(x(t))^\intercal dt, \tag{18c}$$

$$\delta_k(K_i) = \sum_{p=1}^n\sum_{q=1}^m[K_i]_{(p,q)}\delta_k(\mathbf{e}_{pq}), \tag{18d}$$

$$r_k = r_k(Q) + r_k(K_i), \tag{19a}$$

$$r_k(Q) = \int_{t_{x[k]}}^{t_{x[k]}+\tau}x(t)^\intercal Qx(t)dt, \tag{19b}$$

$$r(\mathbf{e}_{pq}) = \int_{t_{x[k]}}^{t_{x[k]}+\tau}x(t)^\intercal\mathbf{e}_{pq}x(t)dt,\ \mathbf{e}_{pq} \in \mathbb{R}^{n\times n} \tag{19c}$$

$$r_k(K_i) = \sum_{p=1}^{n}\sum_{q=1}^{n}[K_i^{\intercal} R K_i]_{(p,q)} r(\mathbf{e}_{pq}), \tag{19d}$$

The K_i dependent terms are (18d) and (19d). If $\mathbf{A}$ in (17a) remains invertible for the recomputed K_i dependent terms, then there is no need for new closed-loop data. Otherwise, new data is needed such that the choice of $x[k]$ measurements ensures $\mathbf{A}$ in (17a) is invertible. Note that in (18b), $u(x) = Fx$ and thus knowledge of F is not required as long as $u(x)$ is accessible and measurable.

IV. Policy Iterations: New Solvability Analysis

In this section, we address the issues raised by Remark 1 and Remark 2. Theorem 1 addresses Remark 2.

Theorem 1. *Let K_i be fixed and such that $A + BK_i$ is Hurwitz. Assume $u(x) = Fx$ drives (1a) such that $x(t) = e^{(A+BF)(t-t_k)}x(t_k)$. Let $\Delta(u, u_i) = Fx - K_i x = L_i x$. It follows that:*

A. *$\forall F$, there exists a* unique *solution $\hat{V}(x)$ to the integral equation*

$$\hat{V}(x(t_k+\tau)) - \hat{V}(x(t_k)) - \int_{t_k}^{t_k+\tau} \Delta(u,u_i)^{\intercal} B^{\intercal} \frac{d\hat{V}}{dx} dt = -\int_{t_k}^{t_k+\tau} (x^{\intercal} Q x + u_i^{\intercal} R u_i)\, dt, \tag{20a}$$

$$\hat{V}(0) = 0, \tag{20b}$$

with integration terms carried over $x(t, t_k, x(t_k), u = Fx)$. The solution is given by $\hat{V}(x) = x^{\intercal} P_i x$, where P_i is defined by (5) and is the unique solution to (8a) for the associated K_i.

B. *$\forall F$, there exists a* nonunique *solution pair $\{\hat{V}(x), \hat{u}(x)\}$ to the integral equation*

$$\hat{V}(x(t_k+\tau)) - \hat{V}(x(t_k)) + 2\int_{t_k}^{t_k+\tau} \Delta(u,u_i)^{\intercal} R \hat{u} dt = -\int_{t_k}^{t_k+\tau} (x^{\intercal} Q x + u_i^{\intercal} R u_i)\, dt, \tag{21a}$$

$$\hat{V}(0) = 0,\ \ \hat{u}(0) = 0, \tag{21b}$$

Proof. Existence of solutions to (20), follows from the fact that $\hat{V}(x) = x^{\intercal} P_i x$ is a solution as it an be differentiated over $x(t, t_k, x(t_k), u = Fx)$ as in (13) then integrated as in (14), where P_i is defined by (5). To show uniqueness, assume there is a solution given by $\hat{V}(x) = x^{\intercal}\hat{P}x$ and substitute it together with $x(t_k + \tau) = e^{(A+BF)\tau}x(t_k)$ in (20a) to get

$$x(t_k)^{\intercal}\left(\mathrm{e}^{(A+BF)\tau}\right)^{\intercal}\hat{P}\mathrm{e}^{(A+BF)\tau}x(t_k) - x(t_k)^{\intercal}\hat{P}x(t_k) - 2x(t_k)^{\intercal}\int_{t_k}^{t_k+\tau}\left(\mathrm{e}^{(A+BF)t}\right)^{\intercal} L_i^{\intercal} B^{\intercal}\hat{P}\mathrm{e}^{(A+BF)t}dt x(t_k) = -x(t_k)^{\intercal}\int_{t_k}^{t_k+\tau}\left(\mathrm{e}^{(A+BF)t}\right)^{\intercal}\left[K_i^{\intercal} R K_i + Q\right]\mathrm{e}^{(A+BF)t}dt x(t_k), \tag{22}$$

where $L = F - K_i$. Since (22) is valid for all $x(t_k)$, we must have

$$\left(\mathrm{e}^{(A+BF)\tau}\right)^{\intercal}\hat{P}\mathrm{e}^{(A+BF)\tau} - \hat{P} - \int_{t_k}^{t_k+\tau}\left(\mathrm{e}^{(A+BF)(t-t_k)}\right)^{\intercal}\left[L_i^{\intercal} B^{\intercal}\hat{P} + \hat{P}BL_i\right]\mathrm{e}^{(A+BF)(t-t_k)}dt = -\int_{t_k}^{t_k+\tau}\left(\mathrm{e}^{(A+BF)(t-t_k)}\right)^{\intercal}\left[K_i^{\intercal} R K_i + Q\right]\mathrm{e}^{(A+BF)(t-t_k)}dt. \tag{23}$$

Since the right-hand side and left-hand side of (23) are smooth and analytic, their Taylor series expansions must be equal. Differentiating (23) once with respect to τ we get the first term of the Taylor series,

$$\left(\mathrm{e}^{(A+BF)\tau}\right)^{\intercal}(A+BF)^{\intercal}\hat{P}\mathrm{e}^{(A+BF)\tau} + \left(\mathrm{e}^{(A+BF)\tau}\right)^{\intercal}\hat{P}(A+BF)\mathrm{e}^{(A+BF)\tau} - \left(\mathrm{e}^{(A+BF)\tau}\right)^{\intercal}\left[L_i^{\intercal} B^{\intercal}\hat{P} + \hat{P}BL_i\right]\mathrm{e}^{(A+BF)\tau} = -\left(\mathrm{e}^{(A+BF)\tau}\right)^{\intercal}\left[K_i^{\intercal} R K_i + Q\right]\mathrm{e}^{(A+BF)\tau}. \tag{24}$$

Setting $\tau = 0$ in (24), this shows that $\hat{P}$ must satisfy

$$(A+BF)^{\intercal}\hat{P} + \hat{P}(A+BF) - L_i^{\intercal} B^{\intercal}\hat{P} - \hat{P}BL_i = -K_i^{\intercal} R K_i - Q. \tag{25}$$

Substitute $L_i = F - K_i$ in (25) and canceling common terms, it follows that $\hat{P}$ must be a solution to $\hat{P}(A + BK_i) + (A + BK_i)^{\intercal}\hat{P} + K_i^{\intercal} R K_i + Q = 0$ which due to the Hurwitz condition on $A + BK_i$ must have a unique solution which is P_i per (8a). Hence, $\hat{P} = P_i$.

Existence of solutions to 21 follows directly from substituting $\hat{u}(x) = -R^{-1}B^{\intercal}P_i$ in 21a to get 20a and then substituting $\hat{V}(x) = x^{\intercal}P_i x$, thus $\{\hat{V}(x) = x^{\intercal}P_i x, \hat{u}(x) = -R^{-1}B^{\intercal}P_i\}$ is a solution pair. It remains to show nonuniqueness which we accomplish by constructing other solutions. Assume there is a solution given by $\hat{V}(x) = x^{\intercal}\hat{P}x$ and $\hat{u}(x) = \hat{K}x$. Substitute it together with $x(t_k + \tau) = e^{(A+BF)\tau}x(t_k)$ in (21a) to get

$$x(t_k)^{\intercal}\left(\mathrm{e}^{(A+BF)\tau}\right)^{\intercal}\hat{P}\mathrm{e}^{(A+BF)\tau}x(t_k) - x(t_k)^{\intercal}\hat{P}x(t_k) + 2x(t_k)^{\intercal}\int_{t_k}^{t_k+\tau}\left(\mathrm{e}^{(A+BF)t}\right)^{\intercal} L_i^{\intercal} R\hat{K}\mathrm{e}^{(A+BF)t}dt x(t_k) = -x(t_k)^{\intercal}\int_{t_k}^{t_k+\tau}\left(\mathrm{e}^{(A+BF)t}\right)^{\intercal}\left[K_i^{\intercal} R K_i + Q\right]\mathrm{e}^{(A+BF)t}dt x(t_k), \tag{26}$$

where $L_i = F - K_i$. Similar to the steps done for (22) and (23), $\hat{P}$ and $\hat{K}$ must satisfy

$$(A+BF)^{\intercal}\hat{P} + \hat{P}(A+BF) + L_i^{\intercal} R\hat{K} + \hat{K}^{\intercal} R L_i = -K_i^{\intercal} R K_i - Q. \tag{27}$$

Conversely, if $\{\hat{P}, \hat{K}\}$ is a solution to (27), then $\{\hat{V}(x) = x^{\intercal}\hat{P}x, \hat{u}(x) = \hat{K}x\}$ is a solution to (21a). To see this, note that (27) can be written as $\dot{\hat{V}} + 2\hat{u}^{\intercal}R\Delta(u, u_i) = -x^{\intercal}Qx - u_i^{\intercal}Ru_i$, where the time derivative is over $x(t, t_k, x(t_k), u = Fx)$, which can then be integrated from t_k to $t_k + \tau$ over the trajectories $x(t, t_k, x(t_k), u = Fx)$ to get (21). Therefore, is sufficient to show that (27) has nonunique solutions.

Equation (27) can be decomposed into the following matrix equations

$$(A+BF)^{\intercal}\hat{P} + \hat{P}(A+BF) = W_1, \tag{28a}$$

$$L_i^{\intercal} R\hat{K} + \hat{K}^{\intercal} R L_i = W_2, \tag{28b}$$

$$-W_1 - K_i^{\intercal} R K_i - Q = W_2, \tag{28c}$$

where (28a) is a Sylvester equation to solve for $\hat{P} \in R^{n\times n}$ and (28b) is a Sylvester-transpose equation to solve for $\hat{K} \in R^{m\times n}$; both Sylvester equations are coupled by (28c). The Sylvester equations correspond to linear maps $\mathcal{T}_1 : P \to W_1$ and $\mathcal{T}_2 : K \to W_2$, thus $W_1 \in \operatorname{Im}\mathcal{T}_1$ and $W_2 \in \operatorname{Im}\mathcal{T}_2$. Existence of solutions to the Sylvester-transpose (28b) is discussed in [17], [18]. To construct solutions to (27) from (28), first choose $W_1 \in \operatorname{Im}\mathcal{T}_1$ such that $W_2 \in \operatorname{Im}\mathcal{T}_2$ via (28c); then, for the chosen W_1 and W_2 solve (28a) and (28b) seperately to find a solution pair $\{\hat{P}, \hat{K}\}$. Conversely, given a solution $\{\hat{P}, \hat{K}\}$ to (27), then $\{\hat{P}, \hat{K}\}$ satisfies (28) for an appropriate W_1 and W_2.

Note that $\forall F$, $\{\hat{P} = P_i, \hat{K} = -R^{-1}B^\intercal P_i\}$ is a solution to (27), thus to 28, resulting in $W_1 = L_i^\intercal B^\intercal P_i + P_i B L_i - K_i^\intercal R K_i - Q$ and $W_2 = -L_i^\intercal B^\intercal P_i - P_i B L_i$ where $L_i = F - K_i$, and where both $W_1 \in \operatorname{Im}\mathcal{T}_1$ and $W_2 \in \operatorname{Im}\mathcal{T}_2$. We can construct additional solutions to 28, and thus to (27), as follows:

a) Common eigenvalue: If $A + BF$ and $-(A + BF)$ have a common eigenvalue λ, choose $W_1 = L^\intercal B^\intercal P_i + P_i B L - K_i^\intercal R K_i - Q$ and thus $W_2 = -L_i^\intercal B^\intercal P_i - P_i B L_i$ which as noted earlier are $W_1 \in \operatorname{Im}\mathcal{T}_1$ and $W_2 \in \operatorname{Im}\mathcal{T}_2$. Cleary, $\hat{K} = -R^{-1}B^\intercal P_i$ solves (28b). Let $w^\intercal(A + BF) = -\lambda w^\intercal$ and $(A+BF)^\intercal v = \lambda v$. It follows that $\hat{P} = P_i + v w^\intercal$ is a solution to (28a) where $v w^\intercal \in \ker \mathcal{T}_1$. Thus $\{\hat{P} = P_i + v w^\intercal, \hat{K} = -R^{-1}B^\intercal P_i\}$ is a solution to (27).

b) No common eigenvalue: If $A + BF$ and $-(A + BF)$ have no common eigenvalue, then $\forall W_1$ there exists a unique solution to (28a). Let $W_1 = -K_i^\intercal R K_i - Q$ and let P_{W_1} be the associated unique solution to (28a), which happens to be also equal to

$$P_{W_1} = \int_{t_0}^{\infty} \left[e^{(A+BF)(t-t_0)}\right]^\intercal \left[Q + K_i^\intercal R K_i\right] e^{(A+BF)(t-t_0)} dt.$$

From W_1, it follows that $W_2 = 0$ and thus a solution to (28b) would be such that $\hat{K} \in \ker \mathcal{T}_2$. Thus $\{\hat{P} = P_{W_1}, \hat{K} = 0\}$ is a solution to (28), and thus to (27).

□

The following results are in relation to the choice of data $x(t_k)$ and $x(t_k + \tau)$ used to solve the linear system (11). In particular we address the data collinearity issue raised in Remark 1 and analyze the data-driven computational scheme to dissect the role of exploitation from that of exploration.

Definition 1. *The* spectrum *of a square matrix A is*

$$\Lambda(A) \triangleq \{\forall \lambda : \det(\lambda I - A) = 0\}.$$

Definition 2. *Let $\sigma(x) = [\sigma_1(x), \dots, \sigma_N(x)]^\intercal$ where $\sigma_i(x) : \mathbb{R}^n \to \mathbb{R}$, and $N \geq 2$. Let $\Gamma = \{a \in \mathbb{R}^{N\times 1} : a_i \neq 0, a_j \neq 0, i \neq j\}$. The set of N functions $\sigma_i(x)$ is* dependent *iff*

$$\exists w \in \Gamma, \forall x \in \mathbb{R}^n : \sigma(x)^\intercal w = 0,$$

and is independent *iff*

$$\forall w \in \Gamma, \exists x \in \mathbb{R}^n : \sigma(x)^\intercal w \neq 0.$$

Lemma 1. *$\exists X = [x(1), \dots, x(N)] \in \mathbb{R}^{n\times N}$ such that $\Phi = [\phi(x(1)), \dots, \phi(x(N))] \in \mathbb{R}^{N\times N}$ is full rank, where $\phi(x(\cdot)) = vec^L(x(\cdot)x(\cdot)^\intercal)$ and $N = \frac{n(n+1)}{2}$.*

Corollary 1. *Let $\phi(x) = vec^L(xx^\intercal)$ where $x \in \mathbb{R}^n$. The set of $N = \frac{n(n+1)}{2}$ functions $\phi_i(x)$ is linearly independent.*

Lemma 2. *Let $N = \frac{n(n+1)}{2}$, and let $\mathbf{A} = [\sigma(x(t_k)) \dots \sigma(x(t_N))]^\intercal$ where $\sigma(x) = [\sigma_1(x), \dots, \sigma_N(x)]^\intercal$ with $\sigma(x(t_k)) = \phi(x(t_k)) - \phi(x(t_k + \tau))$ as shown in (11a). Let $x(t_k + \tau)$ satisfy (3) for a stabilizing K_i. If $\phi_1(x), \dots, \phi_N(x)$ is a linearly independent set, then $\forall \tau : \exists [x(t_1), \dots, x(t_N)] : rank(\mathbf{A}) = N$.*

To explore the cases for which learning along a single state-space trajectory fails requires decomposing (3) in terms of its generalized modes. Due to space limitation, we refer the reader to [21] for an extended discussion of both on-policy and off-policy learning. We limit the discussion here to the following lemma and theorem useful for Section VI.

Lemma 3. *Let A be diagonalizable. Assume $\exists \lambda \in \Lambda(A)$ such that λ is m-fold degenerate. $\forall x(t_0) \in \mathbb{R}^n$, if $\mathbf{A}$ in (11a) is formed from data points along $x(t_k) = e^{A(t_k - t_0)}x(t_0)$, then $rank(\mathbf{A}) < N$.*

Theorem 2. *To avoid data collinearity in (11a), it is necessary to explore a minimum number of $\frac{n(n+1)}{2}$ pairwise distinct state-space trajectories.*

V. Shared Linear Quadratic Regulator

In this section, we leverage policy iterations and the analysis provided in Sections III and IV to synthesize solutions to Problems 1 and 2 posed in Section II.

In Problem 1, the aim is to find $u_a(x)$ that optimizes the human-in-the-loop closed-loop dynamics (1) by minimizing (2). Since the dynamics as seen by $u_a(x)$ is shaped by the human input, we lump the human input with A and rewrite the closed-loop dynamics as follows

$$\dot{x} = A_h + B u_a(x), \tag{29}$$

where $A_h = A + BK_hC_h$ is unknown to the autonomy system. Note that the integrand in (2) is quadratic in x and u_a, thus the underlying ARE is given by

$$0 = PA_h + {A_h}^\intercal P - PBR^{-1}B^\intercal P + Q_h, \tag{30}$$

where $Q_h = Q + C_h^\intercal K_h^\intercal M K_h C_h$. The minimum autonomy intervention policy is given by $u_a(x) = -R^{-1}B^\intercal P x$ where P is the stabilizing solution of (30).

To solve (30) in a data-driven way, we can use either on-policy learning or off-policy learning. In both cases, we require that $A + BK_hC_h$ is Hurwitz. Moreover, the cost function's design parameters M, Q, R, the learning data size N, and the duration of the reward window τ are all required. Finally, access to signals $x(t)$ and $u_h(x)$ as well as knowledge of the input matrix B are all required by the autonomy system.

In on-policy learning of a minimum intervention policy, we let $u_a(x) = u_i(x)$ at each policy iteration thus the closed-loop is changing at each iteration. We initialize $u_0(x) = 0$ since the open-loop is already stable due to A_h being Hurwitz. This is shown in Algorithm 1.

In the off-policy version, let $u_a(x) = 0$ at each policy iteration thus the closed-loop is fixed – the off-policy is u_a and thus $\Delta(u_a, u_i)$ is used in δ_k in (18a). We initialize $u_0(x) = 0$ since A_h is Hurwitz. For brevity, we do not show the full algorithm, but it is along the same lines of the off-policy algorithm shown in Algorithm 2. Note that u_i is floating during iterations, *i.e.* not injected to the plant.

Algorithm 1 Learning Minimum Intervention Policy (**On-Policy**)

function MAIN
  INITIALIZE($\mathbf{0}$,t_0,x_0)
  repeat
    $u_a(x(t)) \leftarrow u_i(x(t))$ ▷ Closed-loop updates $\forall i$
    $\mathbf{A}, \mathbf{b} \leftarrow$ EXPLOITPOLICY ▷ Exploit to gather data
    $W \leftarrow \mathbf{A}^{-1}\mathbf{b}$; $P_i \leftarrow reshape(W)$ ▷ Compute weights
    $K_{i+1} \leftarrow -R^{-1}B^\intercal P_i$; $i \leftarrow i+1$; $u_i(x(t)) \leftarrow K_i x(t)$
  until P_i converges
  return P_i
function INITIALIZE(gain,time,state) ▷ Set u_0
  $K_0 \leftarrow gain$; $u_0 \leftarrow K_0 x(t)$; $t_0 \leftarrow time$; $x(t_0) \leftarrow state$; $i \leftarrow 0$
  Prepare $\mathbf{A}$ and $\mathbf{b}$ in (11) for data.
function EXPLOITPOLICY
  while $(size(\mathbf{A}) < N) \ \vee (cond(\mathbf{A}) < \epsilon)$ **do**
    $r_x, r_{u_h}, r_{u_i} \leftarrow$ EVALUATEREWARD
    Add r_x, r_{u_h}, r_{u_i} to $\mathbf{b}$ in (11b);
    Add $\phi(x(t_k)), \phi(x(t_k+\tau))$ data to $\mathbf{A}$ in (11a)
    NUDGE ▷ To switch to a new trajectory to explore
    $u_a(x) \leftarrow u_i(x(t))$ ▷ Continue exploiting this policy
  return $\mathbf{A}, \mathbf{b}$
function EVALUATEREWARD
  Dynamics is evolving per $x(t) = e^{(A_h+BK_i)(t-t_k)}x(t_k)$.
  $r_x \leftarrow \int_{t_k}^{t_k+\tau} x^\intercal Q x dt$; $r_{u_h} \leftarrow \int_{t_k}^{t_k+\tau} u_h^\intercal(x) M u_h(x) dt$
  $r_{u_i} \leftarrow \int_{t_k}^{t_k+\tau} u_i^\intercal(x) R u_i(x)) dt$
  return r_x, r_{u_h}, r_{u_i} ▷ Return reward
function NUDGE
  $u_a(x) \leftarrow u_i(x(t)) + PRBS$ ▷ Pseudorandom Binary Sequence

Algorithm 2 Learning Optimal Takeover Policy (**Off-Policy**)

function MAIN
  INITIALIZE($u_h(x)$,t_0,x_0)
  $u_a(x(t)) \leftarrow \mathbf{0}$ ▷ Closed-loop fixed $\forall i$
  $\mathbf{A}, \mathbf{b} \leftarrow$ EXPLOITPOLICY ▷ Exploit to gather data
  repeat
    $W \leftarrow \mathbf{A}^{-1}\mathbf{b}$; $P_i \leftarrow reshape(W)$ ▷ Compute weights
    $K_{i+1} \leftarrow -R^{-1}B^\intercal P_i$; $i \leftarrow i+1$; ; $u_i(x(t)) \leftarrow K_i x(t)$
    $\mathbf{A}, \mathbf{b} \leftarrow$ RECOMPUTE($\mathbf{A}$,$\mathbf{b}$)
  until P_i converges
  return P_i
function INITIALIZE(gain or signal,*time*,*state*) ▷ Set u_0
  $u_0(x(t)) \leftarrow u_h(x(t))$; $t_0 \leftarrow time$; $x(t_0) \leftarrow state$; $i \leftarrow 0$
  Prepare $\mathbf{A}$ and $\mathbf{b}$ in (17) for data.
function EXPLOITPOLICY
  while $(size(\mathbf{A}) < N) \ \vee (cond(\mathbf{A}) < \epsilon)$ **do**
    $\delta_k(F), \delta_k(\mathbf{e}_{pq}), r_k(Q), r_k(\mathbf{e}_{pq}) \leftarrow$ EVALUATEREWARD
    $\delta_k(K_i) \leftarrow$ (18d), $\delta_k \leftarrow$ (18a); $r_k(K_i) \leftarrow$ (19d), $r_k \leftarrow$ (19a)
    Add , r_k to $\mathbf{b}$ in (17b)
    Add $\phi(x(t_k)), \phi(x(t_k+\tau))$, δ_k to $\mathbf{A}$ in (17a)
    NUDGE ▷ To switch to a new trajectory to explore
    $u_a(x) \leftarrow \mathbf{0}$ ▷ Continue exploiting this policy
function EVALUATEREWARD
  Dynamics is evolving per $x(t) = e^{(A_h+BF)(t-t_k)}x(t_k)$.
  $\delta_k(F) \leftarrow$ (18b); $\delta_k(\mathbf{e}_{pq}) \leftarrow$ (18c)
  $r_k(Q) \leftarrow$ (19b); $r_k(\mathbf{e}_{pq}) \leftarrow$ (19c)
  return $\delta_k(F), \delta_k(\mathbf{e}_{pq}), r_k(Q), r_k(\mathbf{e}_{pq})$ ▷ Return K_i free parts
function NUDGE
  $u_a(x) \leftarrow \mathbf{0} + PRBS$ ▷ Pseudorandom Binary Sequence
function RECOMPUTE($\mathbf{A}$,$\mathbf{b}$)
  $\forall k, \delta_k(K_i) \leftarrow$ (18d), $\delta_k \leftarrow$ (18a)
  $\forall k, r_k(K_i) \leftarrow$ (19d), $r_k \leftarrow$ (19a)
  if $cond(\mathbf{A}) < \epsilon$ **then**
    $\mathbf{A}, \mathbf{b} \leftarrow$ EXPLOITPOLICY
  return $\mathbf{A}, \mathbf{b}$

In Problem 2, the aim is to find $u_a(x)$ that is optimal after the human operator is removed leaving the autonomy system alone. Thus the underlying ARE is given by

$$0 = PA + A^\intercal P - PBR^{-1}B^\intercal P + Q, \tag{31}$$

The optimal takeover policy is given by $u_a(x) = -R^{-1}B^\intercal P x$ where P is the stabilizing solution of (31). During learning, $u_a = 0$ and $u_0 = u_h$ as shown in Algorithm 2. This initialization differs from the off-policy implementation for the minimum intervention policy learning. Additionally, the off-policy here is $u_h + u_a$, thus $\Delta(u_h + u_a, u_i)$ is used in δ_k in (18a) which is another difference from the minimum intervention case. Lastly, we should point out that the closed-loop dynamics for the first iteration in all three algorithms is the same, thus the off-policy takeover learning can be implemented in parallel to the learning of the minimum intervention policy.

VI. CAR-FOLLOWING EXAMPLE

We show an application of sLQR to a car-following problem in which a car with a parallel autonomy system is to maintain a particular constant spacing from a leading vehicle, and achieve the same speed.

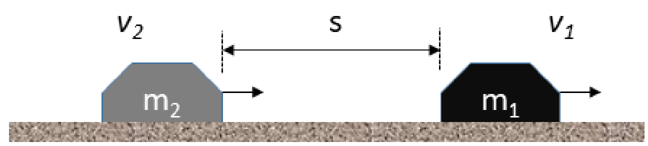

Fig. 2. Car Following.

The error dynamics are adapted from [20]

$$\begin{aligned} \dot{x}_1(t) &= -\frac{\alpha_1}{m_1} x_1(t), \\ \dot{x}_2(t) &= x_1(t) - x_3(t), \\ \dot{x}_3(t) &= -\frac{\alpha_2}{m_2} x_3(t) + \frac{1}{m_2} u, \end{aligned} \tag{32}$$

where $x_1(t) = \tilde{v}_1(t)$, $x_2(t) = \tilde{s}(t)$, $x_3(t) = \tilde{v}_3(t)$ and $u(t) = \tilde{f}_2(t)$. Moreover, $\tilde{v}_1$, $\tilde{v}_2$ are the speed error variables and $\tilde{s}$ is the spacing error variable and $\tilde{f}_2(t)$ is the force error applied to the following car. Let $m_1 = m_2 = 1$ and $\alpha_1 = \alpha_2 = 1$.

We assume that the human operator is applying the following gains $K_h = [0 \ 1 \ -1]$ not known to the autonomy system. Unlike the autonomy system, we assume the human operator has no access to the speed error of the leading vehicle $\tilde{v}_1(t)$, thus

$$C_h = \begin{bmatrix} 0 & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_2 \end{bmatrix}.$$

Note that the A_h matrix of this dynamical system has repeated eigenvalues and is diagonalizable, thus Lemma 3 implies learning cannot happen on the same state-space trajectory. In what follows, we simulate three learning algorithms using the following design parameters $Q = 5\mathbf{I}_3$, $M = 1$, $R = 10$ and $\tau = 0.01$

As seen in the simulations, the exploration needed is minimal and is enough to change current trajectory to a new one to resume learning on. The strength of the nudge is minimal compared to the strength of u_a or u_h. Note also that off-policy learning requires less data as the closed-loop remains fixed, and the gains can be recomputed over the

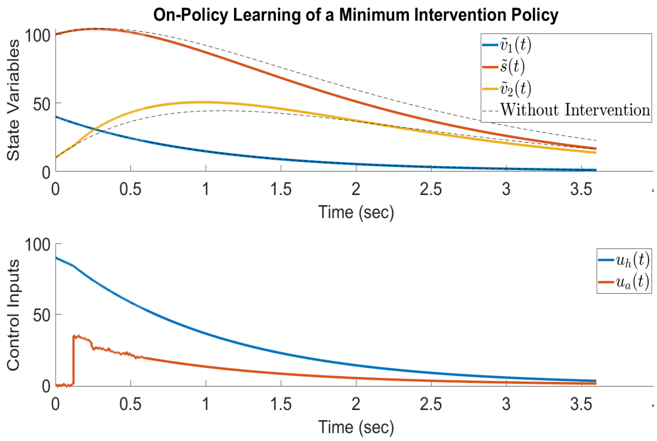

Fig. 3. On-Policy Learning of a Minimum intervention Policy.

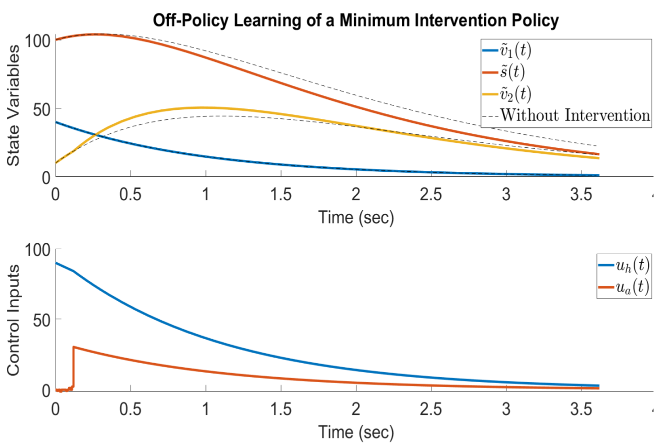

Fig. 4. Off-Policy Learning of a Minimum intervention Policy.

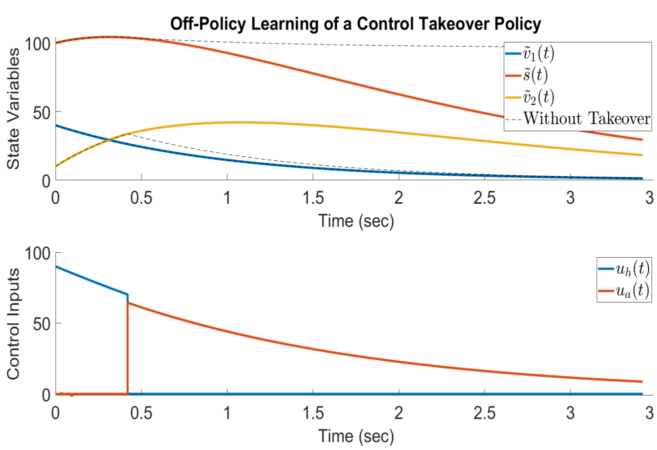

Fig. 5. Off-Policy Learning of a Control Takeover Policy.

trajectory data from the initial iteration. This causes less interference with the human operator and may be more favorable.

VII. Conclusion

The sLQR empowers human operators due to the full access the autonomy system has to the state of the plant. Additionally, the role of exploration ensures minimal interference due to the special requirements of human-in-the-loop parallel autonomy systems. Future work includes verification and validation methods that guard against human policies that may not be stabilizing by leveraging the learned cost-to-go matrices.

References

[1] A. D. Dragan and S. S. Srinivasa, "A policy-blending formalism for shared control," *I. J. Robotic Res.*, vol. 32, no. 7, pp. 790–805, 2013.

[2] S. Javdani, S. S. Srinivasa, and J. A. Bagnell, "Shared Autonomy via Hindsight Optimization," in *Robotics: Science and Systems*, 2015.

[3] S. Reddy, A. D. Dragan, and S. Levine, "Shared Autonomy via Deep Reinforcement Learning," in *Robotics: Science and Systems*, 2018.

[4] S. J. Anderson, S. Peters, T. Pilutti, and K. Iagnemma, "An optimal-control-based framework for trajectory planning, threat assessment, and semi-autonomous control of passenger vehicles in hazard avoidance scenarios," *Int. J. Vehicle Autonomous Systems*, Vol. 8, Nos. 2/3/4, 2010.

[5] S. J. Anderson, S. Karumanchi, and K. Iagnemma, "Constraint-based planning and control for safe, semi-autonomous operation of vehicles," in *Intelligent Vehicles Symposium*, pp. 383–388, 2012.

[6] J. G. Storms, K. Chen, and D. M. Tilbury, "A shared control method for obstacle avoidance with mobile robots and its interaction with communication delay," *I. J. Robotics Res.*, vol. 36, no. 5–7, pp. 820–839, 2017.

[7] W. Schwarting, J. Alonso-Mora, L. Paull, S. Karaman, and D. Rus, "Safe Nonlinear Trajectory Generation for Parallel Autonomy with a Dynamic Vehicle Model," *IEEE Transactions on Intelligent Transportation Systems*, October 2017.

[8] B. Thomsen, A. M. Annaswamy, and E. Lavretsky. "Shared Control Between Human and Adaptive Autopilots", *2018 AIAA Guidance, Navigation, and Control Conference,* 2018.

[9] D. Kleinman, "On an iterative technique for Riccati equation computations", *IEEE Transactions on Automatic Control*, 13, pp. 114-115, 1968.

[10] J. J. Murray, C. J. Cox, G. G. Lendaris, and R. Saeks, "Adaptive dynamic programming," *IEEE Trans. Systems, Man, and Cybernetics, Part C*, vol. 32, no. 2, pp. 140–153, 2002.

[11] M. Abu-Khalaf and F. L. Lewis, "Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach," *Automatica*, vol. 41, no. 5, pp. 779-791, 2005.

[12] D. Vrabie, O. C. Pastravanu, M. Abu-Khalaf, and F. L. Lewis, "Adaptive optimal control for continuous-time linear systems based on policy iteration," *Automatica*, vol. 45, no. 2, pp. 477–484, 2009.

[13] Y. Jiang and Z.-P. Jiang, "Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics," *Automatica*, vol. 48, no. 10, pp. 2699–2704, 2012.

[14] W. Gao and Z.-P. Jiang, "Adaptive Dynamic Programming and Adaptive Optimal Output Regulation of Linear Systems," *IEEE Transactions on Automatic Control*, vol. 61, no. 12, pp. 4164–4169, 2016.

[15] W. Gao, Z.-P. Jiang, and K. Özbay, "Data-Driven Adaptive Optimal Control of Connected Vehicles," *IEEE Trans. Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1122–1133, 2017.

[16] W. Gao, Z.-P. Jiang, F. L. Lewis, Y. Wang, "Leader-to-Formation Stability of Multiagent Systems: An Adaptive Optimal Control Approach," *IEEE Transactions on Automatic Control*, vol. 63, no. 10, pp. 3581-3587, 2018.

[17] H.K. Wimmer, "Roth's theorems for matrix equations with symmetry constraints," *Linear Algebra Appl.*, 199, 357–362, 1994.

[18] A. Wu, Y. Zhang, *Complex Conjugate Matrix Equations for Systems and Control*, Springer Singapore, 2017.

[19] J. Brewer, "Kronecker Products and Matrix Calculus In System Theory". *IEEE Transactions on Circuits and Systems* 25.9: 772-781, 1978.

[20] W. S. Levine, M. Athans, "On the optimal error regulation of a string of moving vehicles," *IEEE Transactions on Automatic Control*, pp. 355-361, 1966.

[21] M. Abu-Khalaf, S. Karaman, and D. Rus, "Shared Linear quadratic Regulation Control: A Reinforcement Learning Approach," in preparation for IEEE transactions., 2019.