Dynamic Actuator Allocation via Reinforcement Learning for Concurrent Plasma Control Objectives

Sai Tej Paruchuri, *Member, IEEE*, Vincent Graber, *Member, IEEE*, Hassan Al Khawaldeh, and Eugenio Schuster, *Member, IEEE*

***Abstract*— Designing a single controller to simultaneously regulate all kinetic and magnetic plasma properties in a tokamak can be difficult, if not impossible, due to the system's complex coupled dynamics. A more viable solution is developing individual algorithms to control one or more plasma properties and integrating them in the plasma control system (PCS) to regulate the target scenario. However, such integration requires an actuator allocation algorithm to convert virtual commands from individual controllers into physical actuator requests (e.g., neutral beam powers) and to arbitrate the competition for available actuators by the different controllers. Existing actuator allocation algorithms rely on solving a static optimization problem at each time instant. Real-time static optimization can be computationally expensive in some instances. Furthermore, static optimization ignores the history of the actuator outputs and the temporal evolution of the actuator constraints. Therefore, dynamic actuator allocation algorithms have been proposed recently as an alternative. These algorithms use ordinary differential equations to describe the relation between virtual commands and physical actuator requests. In this work, a minimax optimization-based dynamic actuator allocation problem is formulated for a certain class of plasma control algorithms. A reinforcement learning (RL)-based algorithm is proposed to solve the optimization and hence the allocation problem. The proposed algorithm is tested using nonlinear simulations.**

***Index Terms*— Actuator allocation, concurrent plasma control, minimax optimization, reinforcement learning (RL).**

I. Introduction

Achieving robust operation in next-generation tokamaks like ITER will rely on the plasma control system's (PCS's) ability to regulate multiple plasma properties around predetermined targets. A common approach in plasma-control design is to synthesize a single control algorithm incorporating more than one control objective [1], [2]. However, as the number of plasma properties to be controlled increases, the complexity of the underlying plasma dynamics increases, and hence, fully integrated control synthesis becomes hard. In particular, the increased dimensionality and complexity of the plasma models make control synthesis intractable.

An alternative to fully integrated control synthesis is to use a two-step approach. The first step consists of synthesizing one or multiple control algorithms. Each algorithm in the first step is designed to prescribe virtual commands depending on the deviation of one or a limited number of plasma variables from the given targets. The second step involves designing an actuator-allocation algorithm that converts the virtual commands into physical actuator requests while managing the competition for different actuators by the plasma controllers. In this step, actuator dynamics and constraints can be incorporated independently of the individual control algorithms. Such a two-step approach can provide multiple advantages over the fully integrated control synthesis approach. For instance, even in the case of a single plasma-control algorithm, the complexity of the plasma control synthesis is vastly reduced due to the decoupling of plasma and actuator dynamics. Additionally, using an actuator allocator makes it easy to account for real-time changes in the actuator availability [3], [4], [5]. Such changes can occur whenever an actuator fails or is redirected for another purpose (such as an electron cyclotron (EC) being repurposed for neoclassical tearing mode (NTM) suppression). Furthermore, the plasma control algorithms can be made tokamak-agnostic. In other words, the plasma control can be designed independently of the available tokamak actuators since they only prescribe virtual commands. Thus, control algorithms designed for and tested on one tokamak can be implemented on another tokamak without modifications (assuming the required measurements or state estimates are available) as long as the controllers are coupled with the tokamak-specific actuator allocators. It is evident from these points that next-generation tokamaks could benefit from a two-step plasma controller design. The implementation of such methodology relies on the development of a robust actuator-allocation algorithm.

Actuator-allocation algorithms could be classified into static or dynamic algorithms. In static algorithms, the relation between the virtual commands prescribed by the control algorithms and the physical actuator requests is assumed to be a linear or nonlinear algebraic equation. A bulk of the actuator-allocation algorithms available in the literature fall under this category [5], [6], [7]. In addition, control algorithms designed to regulate plasma properties simultaneously sometimes implicitly include a static allocation

Manuscript received 4 October 2023; revised 17 February 2024; accepted 27 February 2024. Date of publication 22 March 2024; date of current version 9 December 2024. This work was supported by U.S. Department of Energy, Office of Science, Office of Fusion Energy Sciences, under Award DE-SC0021385 and Award DE-SC0010661. The review of this article was arranged by Senior Editor R. Chapman. *(Corresponding author: Sai Tej Paruchuri.)*

The authors are with the Department of Mechanical Engineering and Mechanics, Lehigh University, Bethlehem, PA 18015 USA (e-mail: saitejp@lehigh.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TPS.2024.3377137.

Digital Object Identifier 10.1109/TPS.2024.3377137

algorithm [8]. Static algorithms generally rely on posing the actuator-allocation problem as an instantaneous optimization problem and solving it. For instance, the algorithm proposed in [6] uses mixed-integer quadratic programming to solve the allocation problem. The algorithm in [7] relies on minimizing a quadratic cost function subject to linear constraints. Another class of actuator-allocation algorithms is the dynamic algorithms. These algorithms model the relation between virtual control commands and the physical actuator requests using differential equations. Since the evolution of the physical actuator requests is related to the virtual commands, dynamic algorithms can be designed to allocate actuator requests that are optimal over the entire tokamak discharge. Furthermore, it is possible to incorporate actuation lags and delays in dynamic algorithms. However, introducing the command-request model based on a differential equation makes the design of dynamic actuation-allocation algorithms challenging. Compared to static algorithms, very few dynamic algorithms are available in the plasma control literature. The dynamic algorithm proposed in [9] uses adaptiveness to meet burn controller objectives while accounting for actuator dynamics.

This work develops a dynamic allocation algorithm that can combine virtual plasma control commands and prescribe physical actuator requests when the number of virtual commands is less than the number of available actuators. The proposed algorithm is designed to be: 1) robust to uncertainties in the plasma and actuator models and 2) optimal over the entire duration of the tokamak discharge. Additionally, the algorithm is designed to be tokamak agnostic as long as the plasma-response models and feedback control laws have a specific structure. The proposed algorithm uses the approach presented in [10] to convert a static command-request model into a dynamic model. As shown in [10], the new dynamic model allows the actuator allocation algorithm to be reformulated as an infinite-horizon minimax optimization problem. Typically, the Hamilton-Jacobi–Isaacs (HJI) equation, a nonlinear partial differential equation (PDE), has to be solved to obtain the solution of the optimization problem. This work uses policy iteration (PI) reinforcement learning (RL) and single-layer neural networks to solve the actuator allocation problem. The proposed approach is tested using nonlinear simulations for two different cases.

This article is organized as follows. Section II discusses the steps involved in developing a dynamic actuator model. Section III reformulates the actuator allocation problem as a minimax optimization problem. The policy-iteration-based methodology used to solve the minimax optimization problem is also presented in Section III. Section IV reviews the results of numerical experiments carried out to validate the proposed algorithm. The conclusions of this work and scope for future extensions are discussed in Section V.

II. Command-Request Relation Modeling

Consider a tokamak scenario in which the control objective is to regulate n plasma states using k physical actuators. Suppose that the evolution of the plasma state vector $\boldsymbol{x} \in \mathbb{R}^n$ is governed by a linear/linearized ordinary differential equation of the form

$$\dot{\boldsymbol{x}} = A\boldsymbol{x} + B\boldsymbol{u} \tag{1}$$

where A and B are the state and input matrices. The vector $\boldsymbol{u} \in \mathbb{R}^m$ represents the virtual commands that are prescribed by o different controllers that are designed to regulate components of $\boldsymbol{x}$. Note that $o \leq m$, which implies that each controller can specify a scalar or vector virtual command. In addition, it is assumed that $m < k$, i.e., the controllers can be designed such that the number of virtual commands is less than the number of physical actuators. To illustrate this framework, consider the problem of plasma energy control by modulating the total power [1]. In this case, the plasma energy is the state, total power is the virtual input, and the individual noninductive drives are the physical actuators.

Let $\boldsymbol{p} \in \mathbb{R}^k$ be the vector of physical actuator requests that are sent to the actuators that are available for control in a given tokamak scenario. The relation between $\boldsymbol{u}$ and $\boldsymbol{p}$ is given by the linear algebraic equation

$$\boldsymbol{u} = G\boldsymbol{p} \tag{2}$$

where $G \in \mathbb{R}^{m\times k}$ is a known matrix with full row rank. In the context of the total plasma energy control problem referred to above, the above equation represents the relation between the total power and the individual auxiliary drive powers. At any given time t, the static physical actuator requests $\boldsymbol{p}_s$ can be computed using the relation

$$\boldsymbol{p}_s = G^{\dagger}\boldsymbol{u} \tag{3}$$

where $G^{\dagger} = G^T(GG^T)^{-1} \in \mathbb{R}^{k\times m}$ is the pseudoinverse of G. However, since the actuator requests computed using the static model in (3) are optimal only at time t, they may not be optimal over the entire plasma discharge [11]. In addition, the actuator requests obtained from (3) may not satisfy the saturation limits.

The goal of dynamic actuator allocation is to choose a "virtual allocation input" $\boldsymbol{\mu}$ that governs the evolution of a "virtual allocation state" $\boldsymbol{v}$ such that the combination of $\boldsymbol{v}$ and $\boldsymbol{p}_s$ is optimal over the period of plasma discharge. The evolution of the virtual state $\boldsymbol{v} \in \mathbb{R}^{k-m}$ is modeled as follows:

$$\dot{\boldsymbol{v}} = \boldsymbol{\mu}. \tag{4}$$

Now, define the dynamic actuator request $\boldsymbol{p}_d$ as follows:

$$\boldsymbol{p}_d = G^{\perp}\boldsymbol{v} + \boldsymbol{p}_s \tag{5}$$

where $G^{\perp} \in \mathbb{R}^{k\times(k-m)}$ is a matrix whose columns form the basis of the nullspace of G, i.e., $GG^{\perp} = \boldsymbol{0} \in \mathbb{R}^{m\times(k-m)}$. Note that (5) satisfies the command-request constraint given in (2). The evolution equation of the "first state" $\boldsymbol{v}$ is given by (4). On the other hand, the evolution equation of the "second state" $\boldsymbol{p}_s$ is obtained by taking the time derivative on both sides of (3), which results in

$$\dot{\boldsymbol{p}}_s = G^{\dagger}\dot{\boldsymbol{u}} \tag{6}$$

where the notation $\dot{(\cdot)}$ denotes the time derivative. Suppose the feedback control law can be written in the form

$$\boldsymbol{u} = -K\boldsymbol{x} \tag{7}$$

where K is the feedback matrix. Substituting (7), (1) and (2) into (6) results in

$$\begin{aligned}\dot{\boldsymbol{p}}_s &= -G^{\dagger}K\dot{\boldsymbol{x}} = -G^{\dagger}K[A\boldsymbol{x} + B\boldsymbol{u}] &(8)\\ &= -G^{\dagger}KBG\boldsymbol{p}_s - G^{\dagger}KA\boldsymbol{x}. &(9)\end{aligned}$$

Define $\boldsymbol{w} = [\boldsymbol{v}^T, \boldsymbol{p}_s^T]^T$. Combining (4), (5), and (9) results in the dynamic allocation model with the state equation

$$\dot{\boldsymbol{w}} = \underbrace{\begin{bmatrix}\boldsymbol{0} & \boldsymbol{0}\\ \boldsymbol{0} & -G^{\dagger}KBG\end{bmatrix}}_{A_a}\boldsymbol{w} + \underbrace{\begin{bmatrix}I\\ \boldsymbol{0}\end{bmatrix}}_{B_a}\boldsymbol{\mu} + \underbrace{\begin{bmatrix}\boldsymbol{0}\\ -G^{\dagger}KA\end{bmatrix}}_{D_a}\underbrace{\boldsymbol{x}}_{\boldsymbol{d}} \quad (10)$$

and the output equation

$$\boldsymbol{p}_d = \underbrace{\begin{bmatrix}G^{\perp} & I\end{bmatrix}}_{C_a}\boldsymbol{w}. \quad (11)$$

Note that the plasma state vector $\boldsymbol{x}$ acts as a disturbance $\boldsymbol{d}$.

III. Actuator Allocation via Optimization

In this section, the problem of computing $\boldsymbol{p}_d$ by choosing $\boldsymbol{w}$ is reformulated as an optimization problem. Given an input policy $\boldsymbol{\mu} : \boldsymbol{w} \mapsto \boldsymbol{\mu}(\boldsymbol{w})$, where $\boldsymbol{w}$ is the state governed by (10), consider a cost function of the form

$$J^{\mu}(\boldsymbol{w}, \boldsymbol{d}) = \int_t^{\infty} e^{-\alpha(\tau - t)}\left(Q(\boldsymbol{p}_d) + \boldsymbol{\mu}^T R\boldsymbol{\mu} - \gamma^2\|\boldsymbol{d}\|^2\right)d\tau \quad (12)$$

where $\boldsymbol{p}_d$ is related to $\boldsymbol{w}$ via (11), $Q : \boldsymbol{p}_d \mapsto Q(\boldsymbol{p}_d) \in \mathbb{R}$ is a scalar valued function that penalizes high values of actuator request $\boldsymbol{p}_d$, $R > 0$ is a positive definite matrix, and $\gamma > 0$ is a constant. Now the optimization problem can be formulated as follows. Determine the optimal policies $\boldsymbol{\mu}^* : \boldsymbol{w} \mapsto \boldsymbol{\mu}^*(\boldsymbol{w})$ and $\boldsymbol{d}^* : \boldsymbol{w} \mapsto \boldsymbol{d}^*(\boldsymbol{w})$ such that

$$J^{\mu^*}(\boldsymbol{w}, \boldsymbol{d}^*) = \min_{\mu}\max_{d} J^{\mu}(\boldsymbol{w}, \boldsymbol{d}). \quad (13)$$

Note that the term $\boldsymbol{d}^*$ represents the disturbance policy that specifies the worst-case disturbance for a given $\boldsymbol{w}$. Thus, the objective of the above optimization problem is to choose the optimal input policy that reduces the cost for the worst-case disturbance. The above problem is typically called minimax optimization or two-player zero-sum game problem and is related to the H_{∞} robust control problem. A unique solution to the above optimization exists [12] if

$$\min_{\mu}\max_{d} J^{\mu}(\boldsymbol{w}, \boldsymbol{d}) = \max_{d}\min_{\mu} J^{\mu}(\boldsymbol{w}, \boldsymbol{d}) \quad (14)$$

and it is given by the following equation:

$$\boldsymbol{\mu}^* = -\frac{1}{2}R^{-1}B_a^T\nabla V^*, \quad \boldsymbol{d}^* = \frac{1}{2\gamma^2}D_a^T\nabla V^*. \quad (15)$$

In the above optimal policies, the term ∇V^* is the gradient of the optimal value function $V^* : \boldsymbol{w} \mapsto V^*(\boldsymbol{w}) \in \mathbb{R}$. The optimal value function is obtained by solving the HJI equation [12]

$$\begin{aligned}Q(C_a\boldsymbol{w}) &+ \nabla V^*(\boldsymbol{w})^T A_a\boldsymbol{w} - \alpha V^*(\boldsymbol{w})\\ &- \frac{1}{4}\nabla V^*(\boldsymbol{w})^T B_a R^{-1} B_a^T \nabla V^*(\boldsymbol{w})\\ &+ \frac{1}{4\gamma^2}\nabla V^*(\boldsymbol{w})^T D_a D_a^T \nabla V^*(\boldsymbol{w}) = 0 \quad (16)\end{aligned}$$

where $\boldsymbol{w} \in \mathbb{R}^{2k-m}$ is an arbitrary vector.

Remark 1: A common choice for Q is

$$Q(\boldsymbol{p}_d) = \boldsymbol{p}_d^T \mathcal{Q}\boldsymbol{p}_d \quad (17)$$

where $\mathcal{Q}$ is a positive semidefinite matrix. With this choice of Q, the solution of the minimax optimization problem discussed above can be obtained by solving the game algebraic Riccati equation (GARE) [12]. Alternatively, the function Q can be designed to incorporate saturation limits. Suppose the saturation limits of the ith actuator are given by the constraint set $[-\bar{\boldsymbol{p}}_d^i, \bar{\boldsymbol{p}}_d^i] \subset \mathbb{R}$, the function Q could be selected as follows:

$$Q(\boldsymbol{p}_d) = \boldsymbol{p}_d^T\mathcal{Q}\boldsymbol{p}_d + \sum_{i=1}^{k}\left(\frac{\boldsymbol{p}_d^i}{\beta\bar{\boldsymbol{p}}_d^i}\right)^{2l} \quad (18)$$

where $l \geq 2$, $\beta \leq 1$ is the safety margin, and $\boldsymbol{p}_d^i$ is the ith component of the vector $\boldsymbol{p}_d$. The higher order terms in the above cost function penalizes large values of $\boldsymbol{p}_d$, thus acting as a "soft constraint." Since only a soft constraint is imposed by the above cost function, it is recommended to use a safety margin β less than 1. For nonsymmetric saturation limits, the discontinuous cost function proposed in [10] could be used.

A. Policy Iteration RL

The HJI equation is a nonlinear PDE, which can be challenging to solve. An alternative approach to arrive at the optimal policies $\boldsymbol{\mu}^*$ and $\boldsymbol{d}^*$ is to use PI, a model-based RL algorithm [12]. PI algorithm begins with initialization of the control and disturbance policies, $\boldsymbol{\mu}_0$ and $\boldsymbol{d}_0$, respectively, and then iteratively implementing two steps: 1) policy evaluation and 2) policy improvement, until convergence. In the policy evaluation step, the Bellman equation

$$\begin{aligned}Q(C_a\boldsymbol{w}) + \nabla V_i{}^T&\left(A_a\boldsymbol{w} + B_a\boldsymbol{\mu}_i + D_a\boldsymbol{d}_i\right)\\ &- \alpha V_i + \boldsymbol{\mu}_i^T R\boldsymbol{\mu}_i - \gamma^2\boldsymbol{d}_i^T\boldsymbol{d}_i = 0 \quad (19)\end{aligned}$$

is solved for V_i. In the Bellman equation, the terms $\boldsymbol{\mu}_i$ and $\boldsymbol{d}_i$ represent the control and disturbance policies obtained in the previous iteration. In the policy improvement step, the value function from policy evaluation step V_i is used to update both the control and disturbance policies using the equations

$$\begin{aligned}\boldsymbol{\mu}_{i+1} &= -\frac{1}{2}R^{-1}B_a^T\nabla V_i &(20)\\ \boldsymbol{d}_{i+1} &= \frac{1}{2\gamma^2}D_a^T\nabla V_i. &(21)\end{aligned}$$

This process of evaluation and improvement is repeated until a stopping criteria is satisfied. The conditions for convergence of V_i to the solution of (16) are discussed in [12]. Note that the Bellman equation is a linear PDE. Thus, the PI algorithm has replaced the problem of solving a nonlinear PDE with that of iteratively solving a linear PDE.

B. NN Parameterization of Value Function

The policy evaluation step discussed above involves solving the Bellman equation. Solving the Bellman equation can be further simplified by introducing a single-layer neural-network (NN) approximation of the value function V. Note that, with a single layer, the neural network approximation can be considered as a linear combination of basis functions. Suppose the value function at the ith iteration can be approximated as follows:

$$V_i(\boldsymbol{w}) = \hat{\boldsymbol{\omega}}_i^T \boldsymbol{\phi}(\boldsymbol{w}) \tag{22}$$

where $\hat{\boldsymbol{\omega}}_i \in \mathbb{R}^N$ is the vector of neural network weights, $\phi(\cdot) := [\phi_1(\cdot), \ldots, \phi_N(\cdot)]$, $\phi_i : \boldsymbol{w} \mapsto \phi_i(\boldsymbol{w})$ is the ith NN basis function ($i = 1, \ldots, N$), and N is the number of neurons. With this approximation, the gradient of the value function can be written as follows:

$$\nabla V_i(\boldsymbol{w}) = \hat{\boldsymbol{\omega}}_i^T \nabla \boldsymbol{\phi}(\boldsymbol{w}) \tag{23}$$

where $\nabla \boldsymbol{\phi}(\boldsymbol{w})$ denotes the Jacobian matrix of $\boldsymbol{\phi}$. Substituting (22) and (23) into the Bellman equation (19) results in a linear algebraic equation of the form

$$a(\boldsymbol{w})\hat{\boldsymbol{\omega}}_i = y(\boldsymbol{w}) \tag{24}$$

where

$$a(\boldsymbol{w}) = \left(A_a \boldsymbol{w} + B_a \boldsymbol{\mu}_i + D_a \boldsymbol{d}_i\right)^T \nabla\phi(\boldsymbol{w})^T - \alpha\phi^T(\boldsymbol{w}) \tag{25}$$

$$y(\boldsymbol{w}) = -Q(\boldsymbol{w}) - \boldsymbol{\mu}_i^T R \boldsymbol{\mu}_i + \gamma^2 \boldsymbol{d}_i^T \boldsymbol{d}_i. \tag{26}$$

Evaluating (24) at M different $\boldsymbol{w}_i$ for $i \in \{1, \ldots, M \geq N\}$, results in M linear equations, which can be written as follows:

$$\underbrace{\begin{bmatrix} a(\boldsymbol{w}_1) \\ \vdots \\ a(\boldsymbol{w}_M) \end{bmatrix}}_{\mathcal{A}} \hat{\boldsymbol{\omega}}_i = \underbrace{\begin{bmatrix} y(\boldsymbol{w}_1) \\ \vdots \\ y(\boldsymbol{w}_M) \end{bmatrix}}_{y}. \tag{27}$$

Thus, with the introduction of the NN approximation of the value function, solving the Bellman equation for V_i simplifies to solving the above linear equation for $\hat{\boldsymbol{\omega}}_i$. Note that the points $\boldsymbol{w}_i$ are selected so that the matrix $\mathcal{A}$ has maximal rank and low condition number.

C. PI Algorithm

The methodology to determine optimal policy for the minimax optimization problem can be summarized as follows.

1) Choose an initial set of allowable stabilizing control and disturbance policies, $\boldsymbol{\mu}_0$ and $\boldsymbol{d}_0$, as defined in [12].
2) Parameterize the value function as shown in (22).
3) *Policy Evaluation:* Solve (27) for $\hat{\boldsymbol{\omega}}_i$ and compute $\nabla V_i(\boldsymbol{w})$ using (23).
4) *Policy Update:* Compute the policies $\boldsymbol{\mu}_{i+1}$ and $\boldsymbol{d}_{i+1}$ using (20) and (21).
5) Return to Step 3) if input and disturbance policy convergence criteria are not satisfied.

The policy obtained from the above algorithm is used to evolve (10) and determine the physical actuator requests using (11), which is computationally inexpensive when compared to real-time static optimization of actuator requests.

Remark 2: Changes in scenario operating conditions can change the matrix G in (2). In such cases, the above algorithm can be implemented in real time to update the policy. Implementation of the above algorithm in the PCS would be fairly simple since only a linear system is solved iteratively.

IV. Numerical Testing of Dynamic Allocator

The results from the simulations carried out to test the effectiveness of the proposed actuator allocation algorithm are reviewed in this section. Two different test cases that deal with the control of particle energies are considered. In the first test case, simultaneous electron and ion energy control is considered. This test case is chosen to highlight the effectiveness of the actuator allocation algorithm for concurrent control purposes. The second test case considers the control of the total plasma energy using multiple actuators. This test case is chosen to highlight how the use of the actuator allocation algorithm simplifies control design. The simulations in both the test cases were carried out using Runge–Kutta solvers.

A. Test Case 1: Electron and Ion Energy Control

The ion energy E_i and electron energy E_e are plasma properties that play a critical role in burn control in reactor-grade tokamaks [9]. They are governed by nonlinear ordinary differential equations of the form

$$\dot{E}_i = -\frac{E_i}{\tau_{E,i}} + \phi_\alpha P_\alpha + P_{ei} + P_{ai} \tag{28}$$

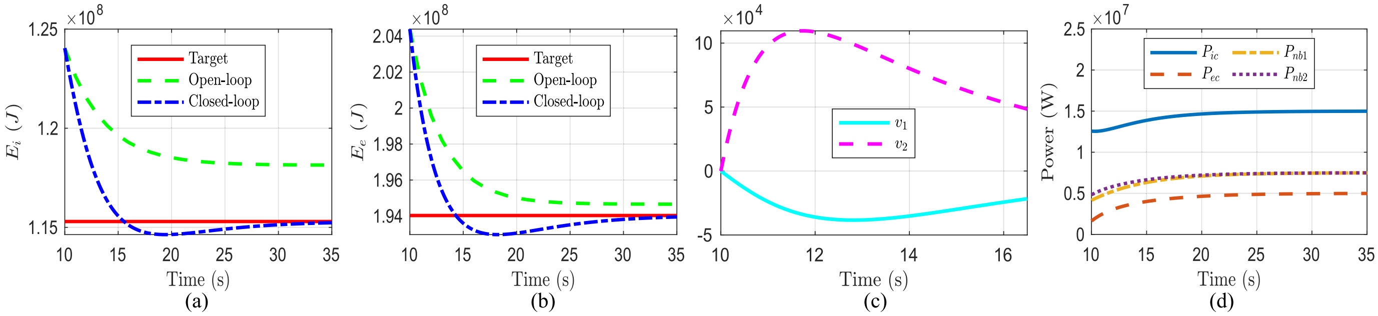
$$\dot{E}_e = -\frac{E_e}{\tau_{E,e}} + (1-\phi_\alpha)P_\alpha - P_{ei} - P_{br} + P_{oh} + P_{ae} \tag{29}$$

where P_α, P_{ei}, P_{br} and P_{oh} are the alpha particle heating power, power exchanged between ions and electrons through collisions, power lost due to bremsstrahlung radiation, and ohmic heating power, respectively. The term ϕ_α is the fraction of alpha particle power deposited into ions. The terms $\tau_{E,i}$ and $\tau_{E,e}$ are ion and electron energy confinement times, respectively, and are proportional to the global energy confinement time τ_E. The global energy confinement time is determined by using the IPB98(y,2) scaling law [13], according to which τ_E depends on the plasma current I_p, line-averaged electron density n_e, and the powers P_α, P_{ei}, P_{br}, P_{oh}, P_{ai}, and P_{ae}. Note that the powers P_α, P_{ei}, P_{br}, and P_{oh} depend on the particle densities and temperatures, which in turn affect the evolution of ion and electron energies. During control synthesis, the terms I_p, n_e, P_α, P_{ei}, P_{br}, and P_{oh} are considered uncontrollable inputs. During a tokamak discharge, any deviations that arise due to variations in these parameters from their nominal values are compensated for by the integral component of the controller described below.

The terms P_{ai} and P_{ae} are the auxiliary power deposited into ions and electrons, respectively, and are considered the virtual commands to be prescribed by the controller. Assuming there are one ion cyclotron, one EC and two neutral beam injectors available for control, the relationship between the virtual commands and the physical actuator requests is

$$\begin{Bmatrix} P_{ai} \\ P_{ae} \end{Bmatrix} = \begin{bmatrix} \eta_{ic}\phi_{ic} & 0 & \eta_{nb1}\phi_{nb} & \eta_{nb2}\phi_{nb} \\ \eta_{ic}\bar{\phi}_{ic} & \eta_{ec} & \eta_{nb1}\bar{\phi}_{nb} & \eta_{nb2}\bar{\phi}_{nb} \end{bmatrix} \begin{Bmatrix} P_{ic} \\ P_{ec} \\ P_{nb1} \\ P_{nb2} \end{Bmatrix} \tag{30}$$

where P_{ic}, P_{ec}, P_{nb1}, and $P_{nb,2}$ are the ion cyclotron, EC, neutral beam 1, and neutral beam 2 powers, respectively, and the terms η_{ic}, η_{ec}, η_{nb1}, and η_{nb2} account for the efficiencies



(a) (b) (c) (d)

Fig. 1. Test case 1: from left to right (a) ion energy evolution in open loop and closed loop, (b) electron energy evolution in open loop and closed loop, (c) virtual allocation states evolution, and (d) physical actuator power/request trajectories.

of these actuators, respectively. In the above equation, ϕ_{nb} and $\bar{\phi}_{nb} = 1-\phi_{nb}$ account for the ion and electron heating fraction of the neutral beam injectors, and ϕ_{ic} and $\bar{\phi}_{ic} = 1-\phi_{ic}$ account for that of the ion cyclotron.

A linear quadratic integral (LQI) controller is designed to regulate the ion and electron energies around given ion energy and electron energy targets, $\bar{E}_i$ and $\bar{E}_e$, respectively. Synthesis of an LQI controller is beyond the scope of this article. The general approach to its synthesis can be found in [1]. To summarize briefly, the LQI design process involves deriving a linear model of the form (1), where the state $\boldsymbol{x}$ is defined as $\boldsymbol{x} := [\tilde{e}^T, \tilde{e}_i^T]^T$ and the virtual commands vector is $\boldsymbol{u} := [P_{ai}, P_{ae}]^T$. Note that the state $\boldsymbol{x}$ contains the ion and electron energy errors $\tilde{e} := [E_i - \bar{E}_i, E_e - \bar{E}_e]^T$, as well as their corresponding integral error components $\tilde{e}_i = \int_0^t \tilde{e} d\tau$. The new linear system is used to formulate a linear quadratic regulator problem, and optimal control theory is used to derive the control law. The resulting control law takes the form given in (7). Since the control model and feedback law have the form shown in Section II, the dynamic actuator algorithm proposed in this article can be used.

The following values were used while carrying out the simulations: $I_p = 15$ MA, $n_e = 10 \times 10^{19}$ m^{-3}, $P_{oh} = 1$ MW, $P_{br} = 15$ MW, $P_{ei} = 1.5$ MW, $P_\alpha = 80$ MW, $\phi_\alpha = 0.2$, $\tau_{E,i} = 1.1\tau_E$, $\tau_{E,e} = 0.9\tau_E$, $\eta_{ic} = 0.9$, $\eta_{ec} = 0.9$, $\eta_{nb1} = 1$, $\eta_{nb2} = 0.8$, $\phi_{nb} = 0.3$, and $\phi_{ic} = 0.8$. The function Q defined in (17) with $\mathcal{Q} = \text{diag}(1.2, 0.9, 1, 1.1)$ was used to carry out the simulations. The term R in (12) was assigned an identity matrix. Other parameters in the cost function (12) were selected as $\alpha = 0.1$ and $\gamma = 40$. Polynomial basis of the second order were used as the NN basis functions, i.e., given $\boldsymbol{w} = [w_1, \ldots, w_{2k-m}]$, the basis were selected as $w_i w_j$, where $i = 1, \ldots, 2k - m$ and $j = i, \ldots, 2k - m$.

In the simulations, the algorithm presented in Section III-C was implemented to determine the optimal policies, and the algorithm converged within six iterations. Fig. 1 shows the simulation results obtained for this case. As evident from the figure, the control algorithm coupled with the dynamic actuator allocation algorithm is able to achieve the control objectives. The figure also shows the virtual allocation states $\boldsymbol{v} = [v_1, v_2]^T$. From (5), it is clear that the proposed optimal dynamic allocator's solution aligns with the static solution only when the virtual states are at the origin. The deviation of the virtual states from the origin shows that the static solution is suboptimal. The physical actuator request values determined by the dynamic actuator allocator are also shown in the figure. As mentioned in Remark 1, the optimal solution corresponding to the specific Q function selected for this case can also be computed through GARE. The policies obtained from PI were compared with those obtained by solving GARE. The Euclidean norm $\|\nabla V(\boldsymbol{w})/2 - P\boldsymbol{w}\|$ for different vectors $\boldsymbol{w} \in \mathbb{R}^{2k-m}$ was determined. This norm is bounded from above by 1.15×10^{-8}. As expected, the PI converged to the optimal solution in this case.

B. Test Case 2: Total Energy Control

This case considers the control of the total plasma energy E in tokamaks with only nonburning plasmas. The total energy is a plasma property related to the normalized beta and is critical to MHD stability. The objective of this test case is to highlight how adding a dynamic actuator algorithm simplifies control synthesis. The total energy is governed by [8]

$$\dot{E} = -\frac{E}{\tau_E} + P_a \tag{31}$$

where P_a is the total auxiliary power and is related to the physical actuator requests through the equation

$$P_a = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{Bmatrix} P_{nb1} & P_{nb2} \end{Bmatrix}^T. \tag{32}$$

In the above model, terms such as alpha particle heating, ohmic heating, and radiation losses are neglected [8]. The terms τ_E, P_{nb1}, and P_{nb2} in (31) and (32) are defined in Test Case 1 presented above. Typically, the actuator powers P_{nb1} and P_{nb2} are selected to match predesigned $\bar{P}_{nb1}$ and $\bar{P}_{nb2}$ so that the total energy is maintained at the predefined target $\bar{E}$. However, model uncertainties and external disturbance can cause the total energy to deviate from the target, and it is critical to drive the total energy back to its target in the shortest time possible. A feedback stabilizing controller can be used to achieve this objective.

The design of a feedback stabilizer involves linearizing (31) around the target $\bar{E}$ and the predetermined auxiliary power $\bar{P}_a = \bar{P}_{nb1} + \bar{P}_{nb2}$. The resulting 1-D linear ODE takes the form

$$\dot{x} = ax + bu \tag{33}$$

which is similar to (1). In the above equation, $x = E - \bar{E}$ and $u = P_a - \bar{P}_a$. The value of a defines the rate at which

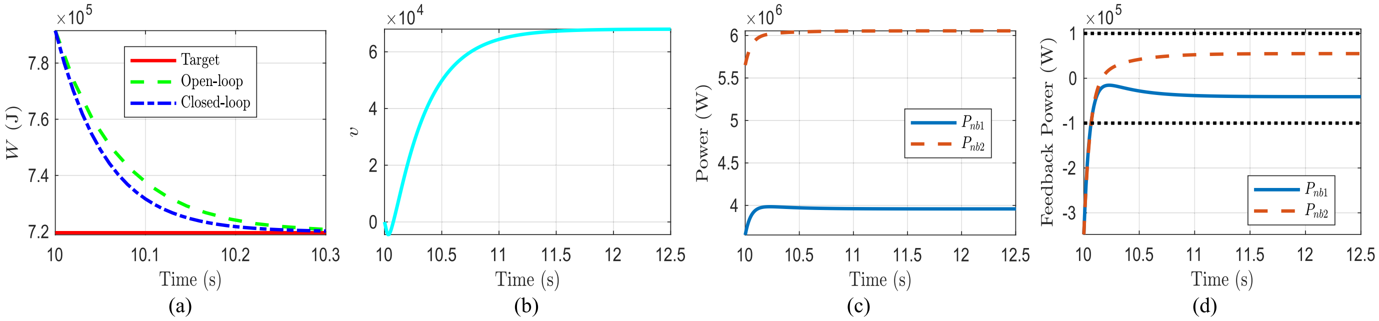


Fig. 2. Test case 2: from left to right (a) total energy evolution in open loop and closed loop, (b) virtual allocation state evolution, (c) physical actuator request trajectories, and (d) feedback component of actuator requests with saturation limits.

the deviation in total energy is stabilized. This rate can be increased by choosing

$$u = -\hat{k}x \tag{34}$$

where $\hat{k}$ is a constant selected such that the rate of convergence of the closed-loop system, defined by $a - b\hat{k}$, matches the desired rate. Since the model and control law have the required structure, the dynamic actuator allocator can be used to prescribe the physical actuator requests P_{nb1} and P_{nb2}. It is certainly possible to design a stabilization algorithm that prescribes the actuator requests directly while satisfying saturation limits. However, with the use of an actuator allocator, the control design becomes trivial and simplifies to choosing a constant $\hat{k}$. The conversion to physical values while satisfying the saturation limits are handled by the allocation algorithm.

The following values were used in the simulations: $I_p = 1.02$ MA, $n_e = 1.05 \times 10^{19}$ m^{-3}. The function Q in (12) was selected as (18) with $\mathcal{Q} = \text{diag}(2, 1)$, $l = 2$, $\beta = 0.9$ and $\bar{\boldsymbol{p}}_d^i = 1 \times 10^5$ for $i = 1, 2$. Other parameters in the cost function (12) were set as $R = 1$, $\alpha = 0.1$, and $\gamma = 40$. The value of $\hat{k}$ in the feedback control law is $\hat{k} = 9.6774$. Since this case considers a more complicated cost function, second-order polynomial basis is insufficient to approximate the value function. Hence, Gaussian radial basis functions centered at 343 equidistant points in a grid of $[-5 \times 10^5, 5 \times 10^5]^3$ were selected as the NN basis functions. For any given center $\boldsymbol{a}_i$ in the grid, the Gaussian basis function is given by $\phi_i(\cdot) = e^{-(\|(\cdot)-\boldsymbol{a}_i\|/2\sigma^2)}$, where σ is a parameter that defines the width of the Gaussian basis function. In the simulations, its value was selected as $\sigma = 2 \times 10^5$. The PI algorithm was implemented for 100 iterations and the resulting value function was used to implement the dynamic actuator algorithm. Note that the algorithm may not converge to the optimal solution since the number of neurons and the choice of NN basis affect the final solution. However, PI provides an effective way to approximate the value function.

The simulation results are shown in Fig. 2. As evident, the total energy stabilizes more quickly in the controlled case. The allocation algorithm modulates the virtual state v as shown in the figure. The total physical actuator powers and their corresponding feedback components are also presented in the figure. The figure shows that the feedback component of the actuator powers violate the saturation constraints at the start of the simulation imposed in the cost function. The function Q defined in (18) only imposes soft constraints. Since no precise combination of physical actuator powers within the saturation limits satisfy the virtual controller commands at the start of the simulation, the constraints are violated. However, after entering the set of physically feasible actuator requests, the values stay within the set. Analyzing the effect of extreme virtual commands that could lead to physically infeasible actuator requests remains a topic of future interest.

V. Conclusions and Future Work

A dynamic actuator allocation algorithm has been developed by deriving a dynamic command-request model, which is then used to formulate a minimax optimization problem. A PI-based algorithm has been presented to solve the optimization problem, particularly for cases that use nonquadratic cost functions. The effectiveness of the actuator allocator has been illustrated using two plasma control test cases. Future studies could focus on relaxing model and plasma control law assumptions, considering cases with more virtual commands than the number of actuators, incorporating hard saturation limits, including actuator lags, introducing actuator management capabilities to handle actuator failures, and testing the proposed algorithm in more complex plasma control scenarios.

Acknowledgment

Sai Tej Paruchuri would like to thank Dr. Patrick Kolaric for his valuable insights and discussions during the course of this research.

References

[1] H. Al Khawaldeh, B. Leard, S. T. Paruchuri, T. Rafiq, and E. Schuster, "Model-based linear–quadratic–integral controller for simultaneous regulation of the current profile and normalized beta in NSTX-U," *Fusion Eng. Des.*, vol. 192, Jul. 2023, Art. no. 113795.

[2] Z. Wang et al., "Implementation and initial testing of a model predictive controller for safety factor profile and energy regulation in the EAST tokamak," in *Proc. Amer. Control Conf. (ACC)*, May 2023, pp. 3276–3281.

[3] E. Maljaars et al., "Simultaneous control of plasma profiles and neoclassical tearing modes with actuator management in tokamaks," in *Proc. 42nd Eur. Phys. Society Conf. Plasma Phys.*, 2015, pp. 1–184.

[4] N. M. T. Vu et al., "Tokamak-agnostic actuator management for multitask integrated control with application to TCV and ITER," *Fusion Eng. Des.*, vol. 147, Oct. 2019, Art. no. 111260.

[5] A. Pajares and E. Schuster, "Integrated control and actuator management strategies for internal inductance and normalized beta regulation," *Fusion Eng. Des.*, vol. 170, Sep. 2021, Art. no. 112526.

[6] E. Maljaars and F. Felici, "Actuator allocation for integrated control in tokamaks: Architectural design and a mixed-integer programming algorithm," *Fusion Eng. Des.*, vol. 122, pp. 94–112, Nov. 2017.

[7] V. Graber and E. Schuster, "Nonlinear adaptive burn control and optimal control allocation of over-actuated two-temperature plasmas," in *Proc. Amer. Control Conf. (ACC)*, Jul. 2020, pp. 1411–1416.

[8] A. Pajares and E. Schuster, “Current profile and normalized beta control via feedback linearization and Lyapunov techniques,” *Nucl. Fusion*, vol. 61, no. 3, Mar. 2021, Art. no. 036006.

[9] V. Graber and E. Schuster, “Nonlinear burn control in ITER using adaptive allocation of actuators with uncertain dynamics,” *Nucl. Fusion*, vol. 62, no. 2, Mar. 2022, Art. no. 026016.

[10] P. Kolaric, V. G. Lopez, and F. L. Lewis, “Optimal dynamic control allocation with guaranteed constraints and online reinforcement learning,” *Automatica*, vol. 122, Dec. 2020, Art. no. 109265.

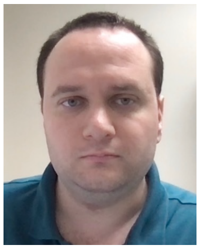
[11] F. L. Lewis, D. Vrabie, and V. L. Syrmos, *Optimal Control*. Hoboken, NJ, USA: Wiley, 2012.

[12] H. Modares, F. L. Lewis, and Z. P. Jiang, “H_∞ tracking control of completely unknown continuous-time systems via off-policy reinforcement learning,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 10, pp. 2550–2562, Oct. 2015.

[13] M. Shimada et al., “Chapter 1: Overview and summary,” *Nucl. Fusion*, vol. 47, no. 6, pp. S1–S17, Jun. 2007.

Sai Tej Paruchuri (Member, IEEE) received the B.E. degree in mechanical engineering from Thiagarajar College of Engineering, Anna University, Madurai, India, in 2014, and the M.S. degree in mathematics and the Ph.D. degree in mechanical engineering from Virginia Tech, Blacksburg, VA, USA, in 2020.

He is a Research Scientist at the Department of Mechanical Engineering and Mechanics, Lehigh University, Bethlehem, PA, USA. His research interests include encompass nonlinear control, adaptive function estimation and control, reproducing kernel Hilbert spaces, data-driven modeling and control, and reinforcement learning. Currently, his research is primarily aimed at addressing control challenges in nuclear fusion and tokamaks.



Vincent Graber (Member, IEEE) received the B.S. degree in mechanical engineering and the Minor degree in energy engineering from Lehigh University, Bethlehem, PA, USA, in 2017. He is currently pursuing the Ph.D. degree in mechanical engineering with the Plasma Control Laboratory, Department of Mechanical Engineering and Mechanics, Lehigh University.

His research interests include cover burn control, actuator management, and control-oriented modeling.

Hassan Al Khawaldeh received the B.S. degree in mechanical engineering from Lehigh University, Bethlehem, PA, USA, in 2021. He is currently pursuing the Ph.D. degree with the Plasma Control Laboratory, Department of Mechanical Engineering and Mechanics, Lehigh University.

His research interests include model-based optimal control, scenario planning, and machine learning.

Eugenio Schuster (Member, IEEE) received the B.Sc. and M.Sc. degrees in electronic engineering from the University of Buenos Aires, Buenos Aires, Argentina, in 1993, the B.Sc. and M.Sc. degrees in nuclear engineering from Balseiro Institute, San Carlos de Bariloche, Argentina, in 1998, and the M.Sc. and Ph.D. degrees in aerospace engineering from the University of California at San Diego, San Diego, CA, USA, in 2000 and 2004, respectively.

He is currently a Professor with the Department of Mechanical Engineering and Mechanics, Lehigh University, Bethlehem, PA, USA. He is an Expert in nuclear-fusion plasma control and leads the Plasma Control Laboratory, Lehigh University. He has been appointed as a Scientist Fellow of Plasma Control by the ITER Organization. Moreover, he has been designated by U.S. Department of Energy (DOE) as an Expert Member of the Integrated Operation Scenarios (IOS) Topical Group within the International Tokamak Physics Activity (ITPA), which he is currently chairing. He was the Leader of the Operations and Control Topical Group, U.S. Burning Plasma Organization (BPO).

Prof. Schuster is the founder chair of the Technical Committee on Power Generation within the IEEE Control Systems Society. He was a recipient of the National Science Foundation (NSF) CAREER Award for his work on “Nonlinear Control of Plasmas in Nuclear Fusion.”