Fully Spiking Linear Quadratic Regulator Control via a Neuromorphic Solver for the Continuous Algebraic Riccati Equation

Graeme Damberger
Centre for Theoretical Neuroscience
University of Waterloo
Ontario, Canada
graeme.damberger@uwaterloo.ca

Omar Alejandro Garcia Alcantara
Electrical and Computer Engineering,
New Mexico State University,
New Mexico, USA.
omargalc@nmsu.edu

Eduardo S. Espinoza
DIEM
CINVESTAV-SECIHTI
Mexico City, Mexico
eduardo.espinoza@cinvestav.mx

Luis Rodolfo Garcia Carrillo
Electrical and Computer Engineering,
New Mexico State University,
New Mexico, USA.
luisillo@nmsu.edu

Terrence C. Stewart
National Research Council of Canada
University of Waterloo
Ontario, Canada
terrence.stewart@nrc-cnrc.gc.ca

Chris Eliasmith
Centre for Theoretical Neuroscience
University of Waterloo
Ontario, Canada
celiasmith@uwaterloo.ca

***Abstract*—A recurrent spiking neural network (SNN) architecture is introduced to dynamically solve the Continuous Algebraic Riccati Equation (CARE), enabling an online, fully spiking implementation of the continuous Linear Quadratic Regulator (LQR). Using a framework that allows dynamical system representation in spiking neurons, the CARE dynamics are embedded directly within a recurrent SNN, yielding a causal controller that computes the LQR solution in real time.**

A Lyapunov-based analysis provides conditions that ensure the stability of both the spiking CARE solver and the resulting closed-loop system. Increasing the neuronal population used to approximate the Riccati differential equation is shown to reduce the tracking error and improve the accuracy of the solution.

Numerical simulations of the yaw dynamics of a constrained quadrotor test platform validate the approach. The proposed controller achieves tracking performance comparable to the ideal non-spiking LQR while preserving the convergent behavior of the CARE within the spiking framework.

***Index Terms*—Spiking Neural Networks, Optimal Control, Continuous Algebraic Riccati Equation, Quadrotor**

I. Introduction

Advances in neuromorphic computing are driving significant innovation within the field of control engineering. Current research in neuromorphic control largely falls into two categories. The first category leverages the learning capabilities of neural networks to map sensory inputs to control commands. For instance, the fully neuromorphic vision-to-control pipeline for controlling a freely flying drone [1], and the Spiking Neural Network (SNN) controller trained using imitation learning from an expert Proportional-Integral-Derivative (PID) controller dataset to control a Crazyflie drone with predictive capabilities [2] are good examples. While these SNN-based controllers can handle complex non-linear mappings, they often lack closed-form stability guarantees and require substantial, data-intensive training.

The second category seeks to mitigate the data-dependency of the first by directly implementing well-established classic control techniques as SNN-based controllers. State-feedback and PID regulator architectures have been mimicked using the Neural Engineering Framework (NEF) to control systems such as the ball and plate platform model [3], a 3-Degrees of Freedom (DoF) [4], and a 6-DoF [5] robotic arm model, all achieving comparable performance to their non-neural counterparts. Other notable work in this category is the Spiking PID with adaptive gains implemented directly on Loihi hardware to control a 1-DoF constrained quadcopter [6]. Additionally, spiking neurons have been used to implement adaptive control by serving as basis functions that complement a sliding mode controller, enabling efficient representation and adaptation [7], [8]. Despite their effectiveness, these SNN-based classic control strategies are inherently not optimal, often relying on heuristic or ad-hoc tuning procedures, and depending on empirical validation rather than rigorous stability proofs.

In classical control theory, the Linear Quadratic Regulator (LQR) addresses these limitations for linear systems [9], [10]. Solving the Algebraic Riccati Equation (ARE) yields an optimal control law that guarantees stability while explicitly balancing the trade-off between state error minimization and control effort [11], a desirable characteristic in neuromorphic control. Recognizing this, several works have attempted to implement LQR controllers using SNNs. The authors in [12] propose a recurrent network of leaky integrate-and-fire (LIF) neurons that emulates an LQR by directly customizing the SNN weight matrices to match the target system model, eliminating the need for training. The authors in [13] utilized the NEF to implement a spiking version of an LQR controller for an underactuated wheeled inverted pendulum, concluding that the noise added by the neural dynamics could help prevent matrices from becoming singular during encoding and decoding processes. In [14], the authors show that the near-linear dynamics of a Leaky Integrate-and-Fire (LIF) neuron can be leveraged to implement LQR control of a time-invariant cart-pole system. Another notable work bridging the gap between optimal control theory and neuromorphic computing is the adaptive path following algorithm of [15]. In this work, an LQR controller is replicated using a neural network endowed with Spike-Timing Dependent Plasticity (STDP), which is modulated by a dopamine-like signal proportional to the vehicle's tracking error. Their key innovation decomposed the LQR control law into sub-functions, implementing them with neural associative maps that use mixed linear (axo-dendritic) and non-linear (axo-axonic) synaptic models to associate input pairs.

While they demonstrated, through empirical cell mapping, that the controller maintains stability and dampens oscillations, the authors explicitly acknowledged that the network's high dimensionality and non-linear structure make formal Lyapunov-based stability proofs largely impractical, revealing the need to simplify the complexity of the proposal without compromising the performance of the controller.

Aside from the lack of Lyapunov-based stability guarantees in [12]–[15], and the need for learning mechanisms in [15], the aforementioned spiking-LQR controllers suffer from a critical shortcoming. The corresponding cost functions were minimized offline by solving an Algebraic Riccati Equation (ARE) to compute the feedback gain, limiting its application to time-invariant systems and requiring standard digital hardware for such computation. These requirements are problematic when targeting fully neuromorphic applications, where the goal is to build fully online, event-based solutions. Moreover, the use of offline solvers is also prone to generating bottlenecks in communication, preventing a truly end-to-end event-driven system.

A. Problem Statement

Despite the potential of SNNs for control, current implementations, whether based on learning techniques or inspired by classic control, compromise on either stability guarantees or rely on heavy training and manual control gains tuning. A principled path forward is to address these limitations by implementing optimal controllers, such as the LQR, directly in spiking hardware. However, this is currently infeasible due to the lack of a causal, online method for solving the Continuous Algebraic Riccati Equation (CARE) within an SNN, a limitation that inherently forces a hybrid digital-neuromorphic architecture. Therefore, the core challenge this work addresses is the absence of a stability-proven spiking neural solver for the CARE that enables an optimal controller fully realized in spiking neurons.

B. Main Contributions

To address these limitations, this manuscript presents a novel Spiking Neural Network architecture that dynamically solves the CARE, enabling an online, spiking LQR controller. Our main contributions are:

1) An online implementation of the continuous LQR controller by dynamically solving the Continuous Algebraic Riccati Equation within an SNN.
2) A Lyapunov-based stability analysis providing conditions of stability of our spiking CARE solver and the resulting closed-loop system.
3) Numerical simulation of a dynamic system model validating the performance and stability of our proposed SNN-LQR controller.

The remainder of this paper is organized as follows: Section II addresses the theory behind the Neural Engineering Framework, which our method uses, along with the basis of Continuous Linear Quadratic Regulators. A Lyapunov stability analysis is derived in Section II-D, relating error from spiking neuron representation to stability margins. In Section III, we present simulation results controlling a dynamic model of the yaw subsystem of a constrained quadrotor test platform and numerically evaluate the condition for stability. Finally, in Section IV we discuss the major implications of our results and lay out our conclusions in Section V.

II. Methods

A. Neural Engineering Framework (NEF)

Neuromorphic computing provides an energy-efficient approach to computation for real-time control by exploiting the temporal sparsity of spiking neurons [16]. Unlike conventional neural networks, spiking neural networks process information through discrete spikes, enabling event-driven computation that is well-suited for continuous-time dynamic systems. These properties make neuromorphic platforms attractive for control applications, including robotic platforms such as quadrotors.

To implement the proposed online Riccati dynamics and neural LQR controller, we use the Neural Engineering Framework (NEF) [17]. The NEF, described as a "neural compiler", provides the methods to represent nonlinear functions and dynamical systems using spiking neurons. The mapping of a dynamic system to spiking networks using the NEF results in a recurrent network with the weights between connections approximating a given function. By following the convention described in [17], we let $\mathbf{x} \in \mathbb{R}^\kappa$ denote a κ-dimensional vector or input to be represented in a neural population of N neurons. Each neuron encodes the input $\mathbf{x}$ into its respective spiking activity as:

$$a_i = G[\alpha_i \langle \mathbf{x}, e_i \rangle + j_i],$$

where α and j are gains and biases, respectively, for a given neuron denoted by i, e describes the preferred direction, and $G[\ ]$ is the nonlinear function which describes the neuron's spiking behavior. A given function $f : \mathbb{R}^l \to \mathbb{R}^q$ is approximated through a linear decoding as:

$$\hat{f}(\mathbf{x}) = \sum_{i=0}^{N} d_i a_i \tag{1}$$

where $d_i \in \mathbb{R}^q$ is the decoder of neuron i. The decoders are solved through the L_2-optimization of:

$$\underset{\mathbf{d}}{\operatorname{argmin}} \frac{1}{|X|} \int_X ||f(\mathbf{x}) - \hat{f}(\mathbf{x})|| d\mathbf{x},$$

over the sampling domain of $X \subset \mathbb{R}^l$. The approximation quality of the reconstruction depends on multiple factors, including the number of neurons. Furthermore, using the above methods of the NEF, any dynamical system in the form:

$$\dot{\mathbf{x}} = f(\mathbf{x}) + g(\mathbf{u}),$$

can be mapped to spiking neural networks through a recurrent connection:

$$f'(\mathbf{x}) = \tau f(\mathbf{x}) + \mathbf{x},$$

and input connection:

$$g'(\mathbf{u}) = \tau g(\mathbf{u}),$$

where f' and g' are the surrogate functions implemented in the neural network using equation (1), in which the dynamics are augmented with the synaptic time constant τ to account for the effect of the synaptic filtering implicit in spiking neural populations.

B. Continuous LQR

A fully continuous-time formulation of the Linear Quadratic Regulator (LQR) is adopted. The plant is modeled as a continuous-time linear time-invariant system of the form:

$$\dot{x} = Ax(t) + Bu(t), \tag{2}$$

where $x \in \mathbb{R}^n$ is the system state, $u \in \mathbb{R}^m$ is the control input, $A \in \mathbb{R}^{n\times n}$ is the state-transition matrix, and $B \in \mathbb{R}^{n\times m}$ is the input matrix. Additionally, we assume the standard LQR conditions that (A, B) is stabilizable, $(A, Q^{1/2})$ is detectable, $Q \in \mathbb{R}^{n\times n}$ is positive semi-definite, and $R \in \mathbb{R}^{m\times m}$ is positive definite. These conditions guarantee the existence of a unique and stabilizing solution to the CARE [18].

The objective of the controller is to minimize the infinite-horizon cost functional:

$$J = \int_0^\infty \left(x(t)^\top Q x(t) + u(t)^\top R u(t) \right) dt. \tag{3}$$

The optimal controller is a state feedback controller in the form:

$$u(t) = -K^* x(t), \tag{4}$$

where $K^* \in \mathbb{R}^{m\times n}$ is obtained from the solution of the CARE. The CARE defines a matrix $P^* \in \mathbb{R}^{n\times n}$ as the unique stabilizing solution to

$$0 = A^\top P^* + P^* A - P^* B R^{-1} B^\top P^* + Q, \tag{5}$$

and the optimal feedback gain is

$$K^* = R^{-1} B^\top P^*. \tag{6}$$

In a traditional LQR, equation (5) is solved offline using direct solvers [19]. However, the neuromorphic implementation introduced in this work solves a dynamical version of the Riccati equation online. Therefore, instead of directly solving the CARE for P^*, we compute the dynamics of $P(t)$ through the continuous-time Riccati Differential Equation (RDE) such that the state converges towards the optimal solution over time. These dynamics are defined as [19]:

$$\dot{P}(t) = -\left(A^\top P(t) + P(t)A - P(t)BR^{-1}B^\top P(t) + Q \right). \tag{7}$$

C. Spiking Implementation

We implement our continuous-time RDE in equation (7) in a SNN using the NEF as discussed in Section II-A and illustrated in Fig. 1. Leveraging the dynamic principle of the NEF, the recurrent connection of the neural ensemble is trained to approximate the augmented CARE dynamics:

$$\dot{P}(t) = f'(P(t)) = \tau f(P(t)) + P(t),$$

where $f(P(t)) = -(A^\top P + PA - PBR^{-1}B^\top P + Q)$ and τ is the synaptic filter time constant.

Since the NEF restricts ensembles of neurons to represent a vector-valued state and $P(t) \in \mathbb{R}^{n\times n}$ for an n-dimensional state system is a matrix, $P(t)$ must be represented as a vector within the neural population. To do this, we flatten the matrix $P(t)$ into a vector $P'(t) \in \mathbb{R}^{n^2}$ by stacking the columns sequentially into a vector form. The ensemble is initialized through an input stimulus, which sets the initial condition of $P(t)$ and is typically chosen as the cost matrix Q from equation (3).

Defining the desired state as x_{des}, a separate neural ensemble represents the state error $E_x = x_{\text{des}} - x$ and the corresponding control gain K. Using the decoded $P(t)$ from the first ensemble, the gain is computed online according to equation (6), producing the control signal as $u = -KE_x$ to be sent to the plant. All neural connections include an exponentially decaying synaptic filter with time constant τ as denoted in Fig. 1. Additionally, a scaling factor S_{scale} is included in some connections to preserve the relative magnitudes of P, K, and x, ensuring the NEF can accurately represent the CARE dynamics and the resulting control law.

D. Stability Analysis

Consider a standard linear time-invariant system from equation (2) with state vector $x(t) \in \mathbb{R}^n$ subject to the state matrix $A \in \mathbb{R}^{n\times n}$ and control input $u(t) \in \mathbb{R}^m$ with control matrix $B \in \mathbb{R}^{n\times m}$. We define the infinite-horizon quadratic cost J as in equation (3), where $Q \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{m\times m}$ are symmetric positive definite matrices. As well, the optimal control law has the form of equation (4) with the gain K^* defined in equation (6), where $P^* \in \mathbb{R}^{n\times n}$ is the unique positive semi-definite solution to the CARE (5). We begin by assuming that the pair (A, B) is stabilizable and (A, Q) is observable, where P will converge to a unique and positive definite solution to equation (5) as P^*. By applying the control

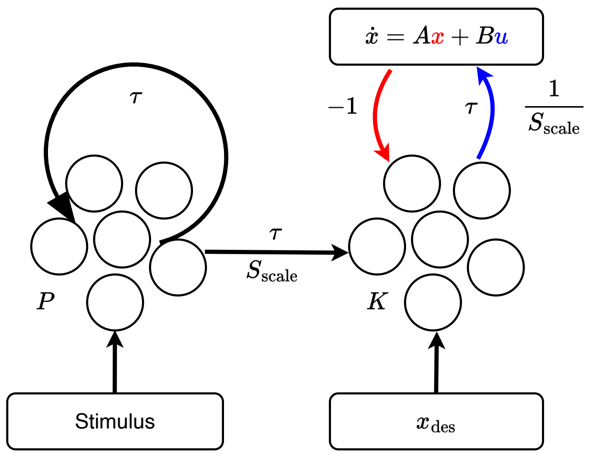

Fig. 1. SNN Architecture. The symbol τ is the synaptic time constant of the connections. S_{scale} represents the transformation value between the connections. Transformations in the NEF can be interpreted as multiplicative scalings. The ensemble P is initialized through an input stimulus, which sets the initial condition of $P(t)$. Additionally, the controlled plant is denoted by the dynamics $\dot{x} = Ax + Bu$

law defined in equation (4) with the optimal converged P^* to the linear system (2), we can rewrite the dynamics as:

$$\begin{aligned} \dot{x} &= Ax + Bu \\ &= Ax - BK^*x \\ &= (A - BK^*)\,x \\ &= A'x, \end{aligned}$$

where we let $A' = A - BK^*$. By the Lyapunov stability analysis as extended from [20], the states x are asymptotically stable if we can find a Lyapunov function $V(x)$ of our state x that is positive-definite and has a time derivative that is negative definite. We start by defining our candidate Lyapunov function:

$$V(x) = x^\top P^* x \tag{8}$$

and taking the time derivative of (8) along the trajectories of the system (2):

$$\begin{aligned} \dot{V}(x) &= \frac{dV}{dt} \\ &= \frac{dV}{dx}\frac{dx}{dt} \\ &= 2x^\top P^* A' x \\ &= x^\top P^* A' x + x^\top A' P^* x. \end{aligned} \tag{9}$$

Therefore, we need to show that the sum of the scalar terms in equation (9) is negative definite, i.e. $x^\top P^* A' x + x^\top A' P^* x < 0$. Substituting equation (6) into equation (5) gives:

$$0 = A^\top P^* + P^* A - P^* BK^* + Q,$$

which we can rewrite as

$$\begin{aligned} P^*A - P^*BK + A^\top P^* - K^{*T}B^\top P^* - P^*BK^* + Q \\ = -P^*BK^* - K^{*T}B^\top P^*. \end{aligned} \tag{10}$$

We then take our new dynamic matrix A' and substitute it into equation (10) for:

$$P^*A' + A'^\top P^* + Q = -K^{*T}B^\top P^*,$$

rewriting as:

$$\begin{aligned} P^*A' + A'^\top P^* &= -K^{*\top}B^\top P^* - Q \\ &= -K^{*\top}RR^{-1}B^\top P^* - Q \\ &= -K^{*\top}RK^* - Q \\ &= -(K^{*\top}RK^* + Q). \end{aligned} \tag{11}$$

We can now substitute equation (11) into the derivative of the candidate Lyapunov function (9) as:

$$\begin{aligned} \dot{V}(x) &= x^\top P^* A' x + x^\top A' P^* x \\ &= x^\top \left(A'^\top P^* + P^* A'\right) x \\ &= -x \left(K^{*\top}RK^* + Q\right) x. \end{aligned} \tag{12}$$

By definition, $Q \geq 0$ and $R > 0$, and therefore we can observe that $K^{*\top}RK^* > 0$. Since the sum of a positive-definite matrix and a positive semi-definite matrix is positive-definite, we can conclude that the term $K^\top RK + Q$ is positive definite. Therefore,

$$\dot{V}(x) = -x^\top(K^{*\top}RK^* + Q)x < 0, \quad \forall x \in \mathbb{R}^n \neq 0$$

Concluding that the conditions for the candidate Lyapunov function in equation (8) are satisfied, and as such the proposed control law in equation (4) renders the dynamics of equation (2) asymptotically stable.

1) Spiking Neural Extension: We now extend the Lyapunov analysis shown in Section II-D to account for the implementation in SNN. For this section, we denote the neural representation of some state variable with the *hat* ($\hat{}$) symbol to account for the differences between an ideal represented and an approximated state. We begin by assuming some functional approximation error $\epsilon(P)$ resulting from a finite representational accuracy of a spiking neural population. Specifically, we assume that this error exists in the representation and dynamic formulation of the matrix P such that the neural representation $\hat{P}$ evolves according to:

$$\dot{\hat{P}} = f(\hat{P}) + \epsilon(\hat{P}). \tag{13}$$

where previously we assumed a converged solution such that $P = P^*$ and $\dot{P}^* = 0$. For the online spiking neural representation, the dynamics of the CARE are computed online, as such $\dot{\hat{P}} \neq 0$. We define the dynamics of $\hat{P}$ as:

$$f(\hat{P}) = -(A^\top\hat{P} + \hat{P}A - \hat{P}BR^{-1}B^\top\hat{P} + Q)$$

with the gain

$$\hat{K} = R^{-1}B^\top\hat{P}.$$

As such, we can rewrite the closed-loop dynamics of equation (2) as

$$\dot{x} = \left(A - B\hat{K}\right) x$$

where

$$\hat{K} = R^{-1}B^\top \hat{P}.$$

By taking the proposed candidate function (8) and considering neural dynamics:

$$V(x) = x^\top \hat{P} x,$$

with time derivative:

$$\dot{V}(x) = \dot{x}^\top \hat{P} x + x^\top \hat{P} \dot{x} + x^\top \dot{\hat{P}} x. \tag{14}$$

We observe that the first two terms of equation (14) are identical to the analysis conducted in Section II-D and can be rewritten as:

$$\dot{x}^\top \hat{P} x + x^\top \hat{P} \dot{x} = -x^\top (Q + \hat{K}^\top R \hat{K}) x.$$

However, the third term of equation (14) is more complex, incorporating the error term defined in equation (13), resulting in:

$$x^\top \dot{\hat{P}} x = x^\top f(\hat{P}) x + x^\top \epsilon(\hat{P}) x.$$

Therefore, adding the residual term $\epsilon(\hat{P})$ limits the time derivative of the candidate Lyapunov function from being strictly negative definite and requires further evaluation. Carrying the third term $x^\top \dot{\hat{P}} x$ through the above derivation results in the following time derivative of the candidate Lyapunov function:

$$\dot{V} = -x^\top (Q + \hat{K}^\top R \hat{K} + \epsilon(\hat{P})) x. \tag{15}$$

By assuming there exists a upper bound $\bar{\epsilon}$ to the error term as follows:

$$||\epsilon(\hat{P})|| \le \bar{\epsilon},$$

we can now update our constraint of stability from (15) to:

$$\dot{V}(x) = -x^\top (Q + \hat{K}^\top R \hat{K}) x + \bar{\epsilon} x^\top x < 0.$$

Since we showed in Section II-D that $-x^\top (Q + \hat{K}^\top R \hat{K}) x$ is negative-definite, the upper bound of the error must not bring $\dot{V}$ to be greater than zero as

$$x^\top (Q + \hat{K}^\top R \hat{K}) x > \bar{\epsilon} x^\top x. \tag{16}$$

Let $\lambda_{\text{min}}(M)$ be the minimum eigenvalue for some given matrix M. We can define a new inequality as:

$$x^\top M x \ge \lambda_{\text{min}}(M) x^\top x, \forall x,$$

letting us rewrite equation (16) as:

$$\lambda_{\text{min}}(M) x^\top x < \bar{\epsilon} x^\top x.$$

Thus, concluding stability occurs at:

$$\bar{\epsilon} < \lambda_{\text{min}}(\hat{K}^\top R \hat{K} + Q). \tag{17}$$

2) Stability Discussion: The stability analysis above shows that the closed-loop system remains asymptotically stable, provided that the functional approximation in the Riccati dynamics, $\bar{\epsilon}$, is sufficiently small. Specifically, the approximation error reduces the stability margin of the LQR controller, acting as a disturbance to the Lyapunov derivative. Because we implement the CARE dynamics and closed-loop controller inside the NEF, the total error can be composed of $\bar{\epsilon} = E_{\text{dist}} + E_{\text{noise}}$, where each term captures a distinct source of deviation from the ideal continuous-time LQR solution. The distortion error E_{dist} arises from the approximation error between the actual and neurally decoded values:

$$E_{\text{dist}} = \frac{1}{2} \int \left\| \mathbf{x} - \sum_{i=1}^{N} d_i a_i(\mathbf{x}) \right\| d\mathbf{x}.$$

The noise-induced error E_{noise} captures the stochastic fluctuations induced by spiking activity:

$$E_{\text{noise}} = \frac{1}{2} \sigma^2 \sum_{i=1}^{N} d_i^2,$$

where d and a are the decoders and activity of a given neuron to an input $\mathbf{x}$ as described in section II-A. Furthermore, σ is the per-neuron spike noise standard deviation, suggested at $\sigma \approx 0.1 a_{\max}$ in [17]. This composition of the error allows us to calculate it to connect tunable parameters, such as network size, to stability guarantees. A potential design policy could consider first choosing an initial neuron count N, then, estimating $\bar{\epsilon} = E_{\text{dist}} + E_{\text{noise}}$ and verifying the stability condition for $\bar{\epsilon} < \lambda_{\min}(\hat{K}^\top R \hat{K} + Q)$, finally, if satisfied, the controller is stable; otherwise, increase the neuron count N.

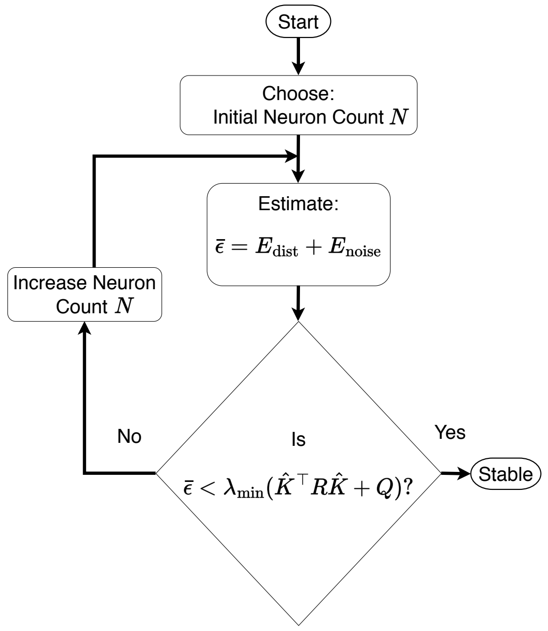
Start
Choose: Initial Neuron Count N
Estimate: $\bar{\epsilon} = E_{\text{dist}} + E_{\text{noise}}$
Increase Neuron Count N
No
Is $\bar{\epsilon} < \lambda_{\min}(\hat{K}^\top R \hat{K} + Q)$?
Yes
Stable

Fig. 2. Design workflow for ensuring closed-loop stability in the proposed spiking LQR controller. The diagram outlines the steps for determining the total neuron count required, evaluating distortion and noise error for stability margins.

This practical design procedure for selecting network parameters that satisfy the derived stability conditions is illustrated in Fig. 2. This provides a quantitative link between neuromorphic resource allocation and closed-loop stability.

III. Simulation Results

A. Experiential setup

The target system to control is the dynamic model of the yaw subsystem of the 3 DOF Hover System from Quanser [21]. It consists of a 4-rotor dynamic testbed constrained at the center of mass. We are interested in the yaw rotational motion about its body axis b_3 described by the angle ψ. The free-body diagram of the 3 DOF Hover System is illustrated in Fig. 3

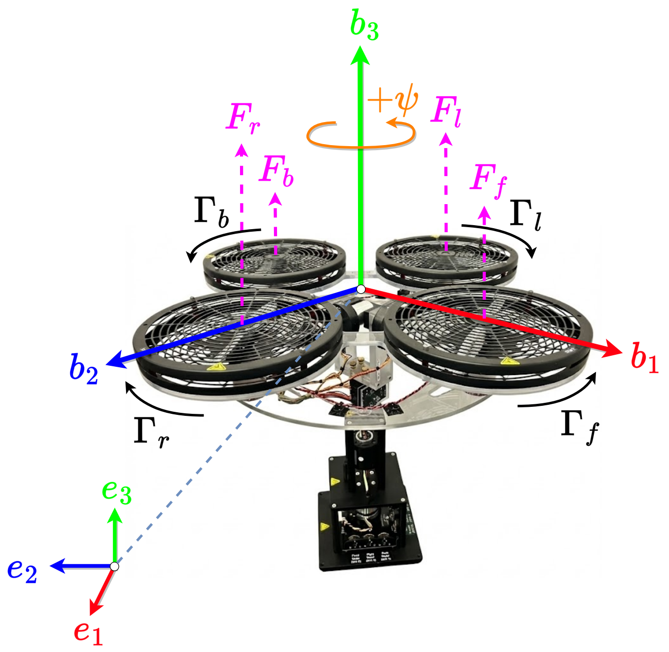

Fig. 3. 3 DoF Hover system diagram. Positive rotations are defined with the right-hand convention.

The following assumptions are made to facilitate the development of the dynamic model for control.

Assumption 1: Principal moment of inertia about the b_3 axis is known.

Assumption 2: Yaw angular position and velocity w.r.t. the inertial frame are measured with sufficient precision.

Assumption 3: Motor dynamics are considered linear.

The differential equation for the yaw rotational motion can be expressed as:

$$\begin{aligned} J_{33}\ddot{\psi} =& \Gamma_l + \Gamma_r - \Gamma_f - \Gamma_b \\ =& k_t\left(V_r + V_l\right) - k_t\left(V_f + V_b\right) \end{aligned}$$

where k_t is the thrust-torque constant, J_{33} is the principal moment of inertia about the b_3 axis, V_f, V_b, V_l, V_r are the front, back, left and right motor voltage, respectively, and Γ_l, Γ_r, Γ_f, Γ_b are the torques generated by the left, right, front and back rotors, respectively. Defining the state vector $x^\top = \begin{bmatrix} \psi & \dot{\psi} \end{bmatrix}$, the output vector $y^\top = \begin{bmatrix} \psi & \dot{\psi} \end{bmatrix}$ and the control vector $u^\top = \begin{bmatrix} V_f & V_b & V_r & V_l \end{bmatrix}$ the corresponding state-space representation is given by:

$$\dot{x}(t) = Ax(t) + Bu(t), \quad y(t) = Cx(t) + Du(t) \tag{18}$$

with the matrices A, B, C and D defined as:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -\frac{k_t}{J_{33}} & -\frac{k_t}{J_{33}} & \frac{k_t}{J_{33}} & \frac{k_t}{J_{33}} \end{bmatrix}$$
$$C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad D = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

where $k_t = 0.0036$ N m/V and $J_{33} = 0.1104$ kg m^2.

B. Results

We begin by demonstrating the performance of the proposed neural LQR controller on the simplified yaw dynamics described by equation (18). The control task requires the state to track a sinusoidal reference signal, $x_{\text{des}} = 0.5\sin(0.1\pi t)$, along with its corresponding time derivative.

The evolution of the CARE state matrix $P(t)$ over time is shown in Fig. 4. The entries of $P(t)$ converge smoothly to their optimal steady state values P^*, indicating that the neural ensemble accurately represents and computes the solution to the CARE.

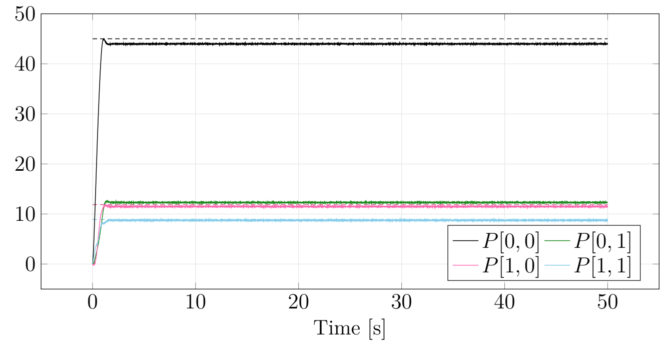

Fig. 4. Time evolution of the entries of the neural Riccati matrix $P(t)$ is represented by solid lines (—,—,—,—). Dashed lines (- - -,- - -,- - -,- - -) indicate the optimal steady-state values P^*.

Fig. 5 illustrates the tracking behavior of the controlled state $x(t)$. The reference trajectory x_{des} is accurately tracked by the actual state under the neural LQR controller, with small errors arising from the inherent approximation errors induced by the spiking neural representation. The Fig. 5 also includes the ideal trajectory $x_{\text{Ideal}} = \begin{bmatrix} \psi_{\text{Ideal}} & \dot{\psi}_{\text{Ideal}} \end{bmatrix}$, for comparison, simulated in the absence of spiking neural networks. We show x_{Ideal} to demonstrate the performance of an ideal LQR controller, such that we can compare tracking performance against the neuromorphic free approach.

Finally, the stability of the controller is assessed in Fig. 6, which compares the minimum eigenvalue $\lambda_{\min}(Q + K^\top RK)$ against the threshold $\bar{\epsilon}$ derived from the stability condition in equation (17). This result confirms that $\lambda_{\min}(Q+K^\top RK) > \bar{\epsilon}$

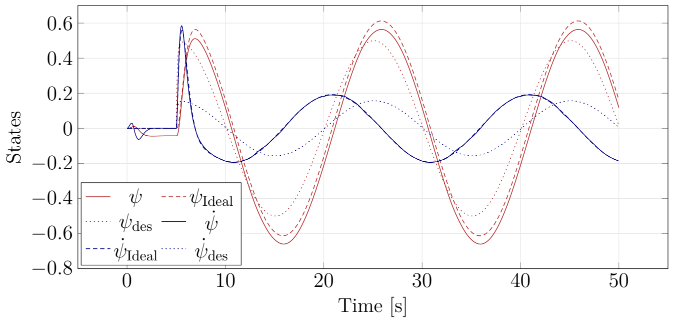

Fig. 5. Tracking performance of the state $x(t)$ denoted by the solid lines (—,—) under neural LQR control. The reference x_{des} is denoted by the dotted lines (⋯,⋯), and the ideal state x_{Ideal} in the absence of spiking neural networks is denoted by the dashed lines (---,---).

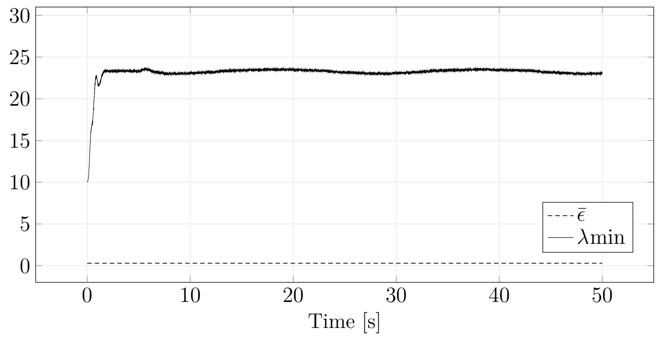

Fig. 6. Minimum eigenvalue $\lambda_{\min}(Q + K^{\top}RK)$ over time demonstrating stability with respect to the threshold $\bar{\epsilon}$.

throughout the simulation, verifying that the closed-loop system remains stable under a neural implementation.

Table I summarizes the effect of increasing the number of neurons in the spiking implementation of the controller in both populations. As expected from the NEF, increasing the number of neurons increases the representational capacity of the ensemble, leading to improvement in performance. Tracking RMSE decreases sharply with diminishing returns between 500 to 1000 neurons. A similar trend is observed in the Riccati state approximation, showing that a higher neuron count lets the neural solution approach the optimal solution. The stability margin derived from the bound in equation (17) with $\bar{\epsilon}$ also improves with neuron count. Noting that with 100 neurons, stability can not be guaranteed as the minimum eigenvalue $\lambda_{\min}$ must evolve dynamically as P converges. It can be observed in Fig. 6 that the value of $\lambda_{\min}$ starts with a value less than the error upper bound $\bar{\epsilon}$ from Table I for 100 neurons. Overall, larger ensembles reduce error from noise and distortion, reducing the representation error.

IV. Discussion

In this work, we propose a fully spiking LQR controller by implementing the CARE in recurrent spiking dynamics. We validate the proposed controller through a stability analysis, concluding with a condition of stability, connecting tunable parameters to performance. The results depicted in Fig. 5 indicate that the proposed controller exhibits a similar performance as with the ideal case that has no spiking neurons. The conditions provided by the stability analysis are validated in Fig. 6, resulting in the convergence of the CARE, illustrated in Fig. 4.

TABLE I
Performance metrics for varying ensemble sizes in both neural ensembles

Number of Neurons	Tracking RMSE	P RMSE	$\bar{\epsilon}$	$\lambda_{\min,SS}$
100	15.815	38.141	14.451	26.451
500	0.453	6.022	3.233	28.719
1000	0.178	4.185	1.493	24.841
5000	0.146	2.012	0.301	24.078

The spiking implementation of the continuous-time LQR controller successfully reproduces the behaviour of the ideal (non-spiking) controller. Figure 5 shows that the state trajectory closely tracks the desired reference signal, with the error decreasing as the number of neurons increases. The CARE matrix $P(t)$ converges approximately to the optimal steady-state solution P^*, see Fig. 4, demonstrating that the recurrent spiking neural network dynamics effectively approximate the true CARE dynamics.

Several limitations of the current approach should be acknowledged. Firstly, the current method of solving the CARE assumes time-invariant dynamics on A and B from the plant. Extending this architecture towards time-varying dynamics is feasible; however, it would require updates to the proposed controller and an extended stability analysis. Secondly, scaling to higher dimensions is challenging, specifically because the CARE requires representing and updating a full matrix $P \in \mathbb{R}^{n\times n}$. Representing n^2 variables in ensembles is feasible for a reasonable n, but becomes challenging for higher-dimensional systems, as the computational load scales quadratically with the state dimension.

V. Conclusions

This work presents a step towards optimal control implemented entirely in spiking neural hardware by introducing a spiking neural solver for the Continuous-Time Algebraic Equation. By embedding the CARE dynamics directly within recurrent spiking neural networks, we enable an online, fully spiking implementation of the LQR controller without the need for additional digital computation. The stability analysis conducted in this work establishes clear conditions under which the spiking CARE remains stable, providing a theoretical foundation for designing controllers in spiking neural networks. We demonstrate that the proposed controller achieves tracking performance comparable to the ideal non-spiking implementation, while retaining converging dynamics of the CARE. Together, these contributions highlight both the feasibility and performance of solving optimal control problems within neuromorphic frameworks.

Future research directions include several key extensions. Develop high-dimensional matrix representations, using structured representations such as low-rank or block-diagonal factorization encodings in the NEF. Extending the solving of the CARE to time-varying systems, allowing for online updating of the controller gain as dynamics evolve. Finally, deployment on neuromorphic hardware and interfacing with physical systems to evaluate energy consumption and real-time performance.

References

[1] F. Paredes-Vallés, J. J. Hagenaars, J. Dupeyroux, S. Stroobants, Y. Xu, and G. C. H. E. de Croon, "Fully neuromorphic vision and control for autonomous drone flight," *Science Robotics*, vol. 9, no. 90, p. eadi0591, 2024.

[2] S. Stroobants, C. De Wagter, and G. C. H. E. de Croon, "Neuromorphic attitude estimation and control," *IEEE Robotics and Automation Letters*, vol. 10, no. 5, pp. 4858–4865, 2025.

[3] O. A. García A., D. C. Arana, E. S. Espinoza, L. R. García Carrillo, A. O. Cordero, and A. T. Sornborger, "Spiking neural network-based control applied to an underactuated system," in *2023 20th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, pp. 1–6, 2023.

[4] D. Marrero, J. Kern, and C. Urrea, "A novel robotic controller using neural engineering framework-based spiking neural networks," *Sensors*, vol. 24, no. 2, p. 491, 2024.

[5] Y. Zaidel, A. Shalumov, A. Volinski, L. Supic, and E. Ezra Tsur, "Neuromorphic NEF-based inverse kinematics and PID control," *Frontiers in Neurorobotics*, vol. 15, p. 631159, 2021.

[6] R. K. Stagsted, A. Vitale, J. Binz, A. Renner, L. B. Larsen, and Y. Sandamirskaya, "Towards neuromorphic control: A spiking neural network based PID controller for UAV," in *Robotics: Science and Systems*, (Corvalis, OR, USA), July 2020.

[7] T. DeWolf, T. C. Stewart, J.-J. Slotine, and C. Eliasmith, "A spiking neural model of adaptive arm control," *Proc. R. Soc. B*, vol. 283, no. 1843, p. 20162134, 2016.

[8] G. Damberger, K. Simone, C. Datta, R. E. Kaundinya, J. Escareno, and C. Eliasmith, "Biologically-inspired representations for adaptive control with spatial semantic pointers," in *2025 Neuro Inspired Computational Elements Conference (NICE)*, pp. 1–10, 2025.

[9] A. A. Alblooshi, I. Hafez, and R. Dhaouadi, "An improved hybrid mrac–lqr control scheme for robust quadrotor altitude and attitude regulation," *Drones*, vol. 9, p. 814, Nov. 2025.

[10] G. E. Setyawan, W. Kurniawan, and A. C. L. Gaol, "Linear quadratic regulator controller (lqr) for ar. drone's safe landing," in *2019 International Conference on Sustainable Information Engineering and Technology (SIET)*, pp. 228–233, 2019.

[11] F. L. Lewis, D. Vrabie, and V. L. Syrmos, *Optimal control: linear quadratic methods*. John Wiley & Sons, 2012.

[12] R. Ahmadvand, S. S. Sharif, and Y. M. Banad, "Neuromorphic robust framework for concurrent estimation and control in dynamical systems using spiking neural networks," *arXiv preprint*, 2024. Available: https://arxiv.org/abs/2307.07963.

[13] J. A. Juárez-Lora, J. H. Sossa Azuela, V. H. Ponce-Ponce, E. Rubio-Espino, and R. Barrón Fernández, "Spiking neural network implementation of lqr control on underactuated system," *International Journal of Combinatorial Optimization Problems and Informatics*, vol. 13, pp. 36–46, Aug 2022.

[14] S. Banerjee, L. Gava, A. Rounak, and V. Pakrashi, "Benchmarking spiking neurons for linear quadratic regulator control of multi-linked pole on a cart: from single neuron to ensemble," 2025. arXiv.

[15] J. Perez Fernandez, M. Alcazar Vargas, J. A. Cabrera Carrillo, J. J. Castillo Aguilar, and B. Shyrokau, "Path-following control using spiking neural networks associative maps," *Robotics and Autonomous Systems*, vol. 193, p. 105077, 2025.

[16] T. DeWolf, P. Jaworski, and C. Eliasmith, "Nengo and low-power ai hardware for robust embedded neurorobotics," *Frontiers in Neurorobotics*, 2020.

[17] C. Eliasmith and C. H. Anderson, *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. MIT Press, 2003.

[18] J. P. Hespanha, *Linear Systems Theory*. USA: Princeton University Press, 2 ed., 2018.

[19] D. e. Kirk, *Optimal Control Theory: An Introduction*. USA: Dover Publications, Inc., 1 ed., 2004.

[20] H. K. Khalil, *Nonlinear Control*. USA: Pearson Education, Inc, 1 ed., 2015.

[21] Quanser Inc., *3-DOF Hover*, 2024. Product Datasheet, Accessed: Dec. 1, 2025.