

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

Graeme Damberger¹, Omar Alejandro Garcia Alcantara², Eduardo S. Espinoza³, Luis Rodolfo Garcia Carrillo², Terrence C. Stewart⁴, Chris Eliasmith¹

1. Centre for Theoretical Neuroscience, University of Waterloo
2. New Mexico State University
3. Center for Research and Advanced Studies of the National Polytechnic Institute
4. National Research Council of Canada, University of Waterloo



Presentation Overview

- Motivation and background
- Problem statement and contributions
- Methods
- Simulation and results
- Discussion and conclusion

Motivation and Background

Neuromorphic Control

- Control theory
 - Regulate the behavior of dynamic systems
 - Example: Proportional Integral Control
- Neuromorphic systems
 - Perform computation on spiking neurons
 - Low power, continuous computation
 - Model biological behaviour

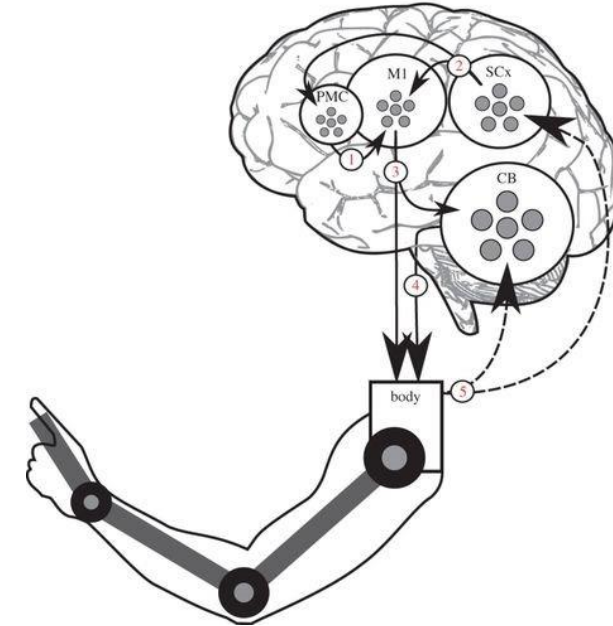


Figure: An overview of the REACH model, shown controlling a three-link arm.

Limitations of Current Controllers

- Learning based controllers
 - Learn mapping from sensory input to control action
 - Dependent on large datasets
 - Limited theoretical guarantees
- Classical controllers
 - Map classical controllers to spiking networks
 - Not optimal, empirically validated
- Visual: Some control performance plot or some comparison between empirical and theoretical validation

Linear Quadratic Regulators (LQR)

- Optimal control algorithm for linear systems
 - State x with dynamic matrices A, B
 - Control input u
- Minimizes a cost function
 - Balance control effort and state error
 - Weighting matrices Q and R
- Gain obtained by solving the Algebraic Riccati Equation (ARE)
- Guarantees closed loop stability

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

$$J = \int_0^{\infty} (x(t)^{\top} Q x(t) + u(t)^{\top} R u(t)) dt$$

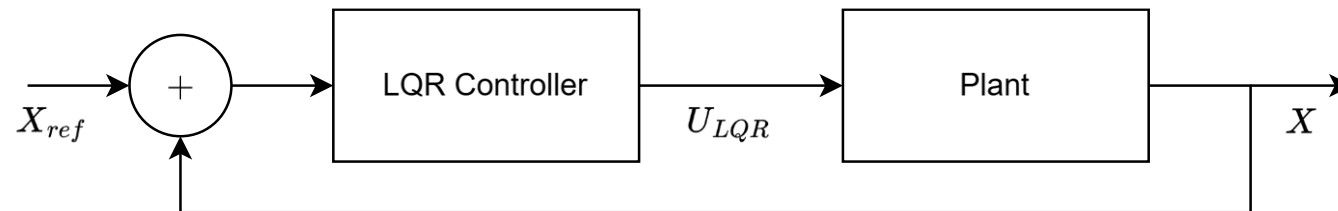


Figure: Closed loop control architecture with LQR controller

Problem Statement and Contributions

Problem Statement

- Current neuromorphic controllers face key trade offs
 - Rely on manual tuning or large data sets
 - Often lack stability guarantees
- LQR provides a solution
 - Optimal control with stability guarantees
 - Requires solving CARE, often offline through digital solvers
- No causal, online spiking methods for the CARE, and subsequently the LQR

Contributions

- Online LQR
 - Dynamically solve the CARE in a spiking neural system
- Stability analysis
 - Lyapunov based conditions ensuring closed loop stability with spiking solver
- Simulation validation
 - Demonstration controller, validating system performance and stability

Methods

Neural Engineering Framework

- State representation
 - Neurons encode vectors as spiking activity a
 - Gain α , bias j , activation function G
- Function approximation
 - Linear decoding reconstructs functions $f(x)$ of the state
 - Decoders d
- Dynamics with recurrent connections
 - Map continuous time system to neurons

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

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

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

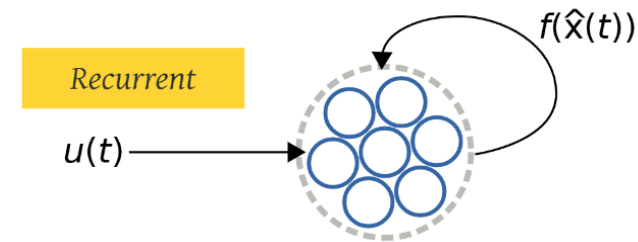


Figure: Dynamics in recurrent connection implemented using the NEF

Continuous LQR and CARE

- Optimal gains obtained from the Continuous Algebraic Riccati Equation (CARE)

- Riccati matrix P

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

- Introduce the LQR and CARE in continuous time

- Solve CARE online as a dynamic system

$$\dot{P}(t) = - (A^\top P(t) + P(t) A - P(t) B R^{-1} B^\top P(t) + Q)$$

- If state converges, P provides the optimal gain K

- Trade off between control and tracking

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

Spiking Implementation

- Encode Riccati matrix as neural ensemble
- Use NEF decoders to approximate the CARE dynamics
 - Implement dynamics through a recurrent connection

$$f(P(t)) = -(A^T P + PA - PBR^{-1}B^T P + Q)$$

- Neural state evolves until convergence
- Extract gains for state feedback controller

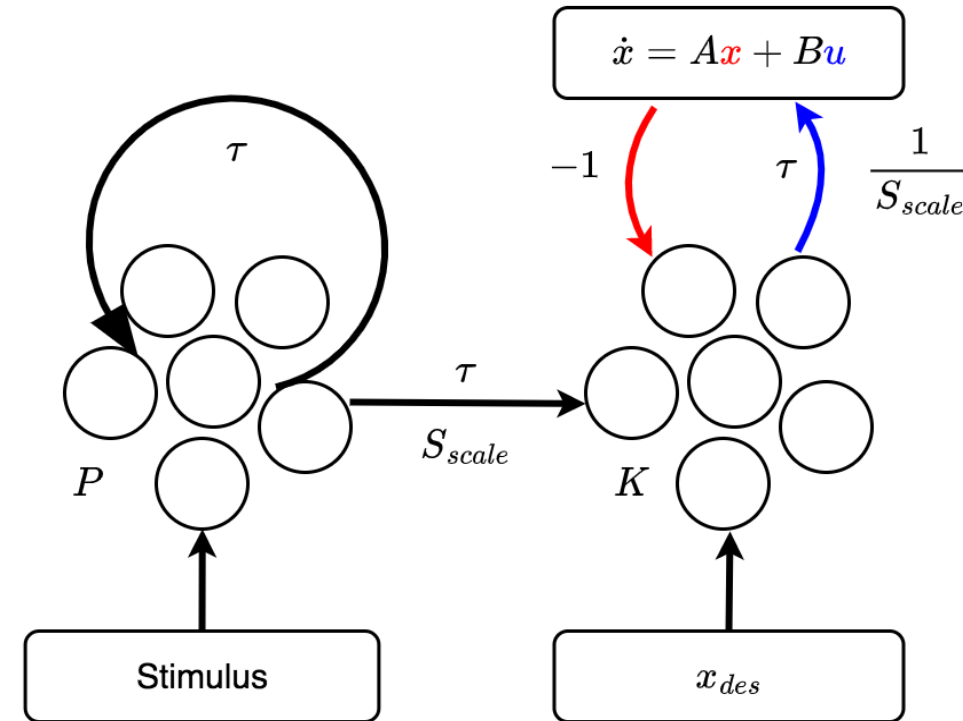


Figure: Spiking implementation of control architecture. The symbol τ is the synaptic time constant and S_{scale} is a linear transform value between connections.

Stability Analysis

- LTI system with optimal control from CARE
 - State x
- Choose Lyapunov candidate function
 - Measures energy of the system

$$V(x) = x^\top P^* x$$

- Find derivative
 - Show derivative is less than zero
 - Decreases along all trajectories

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

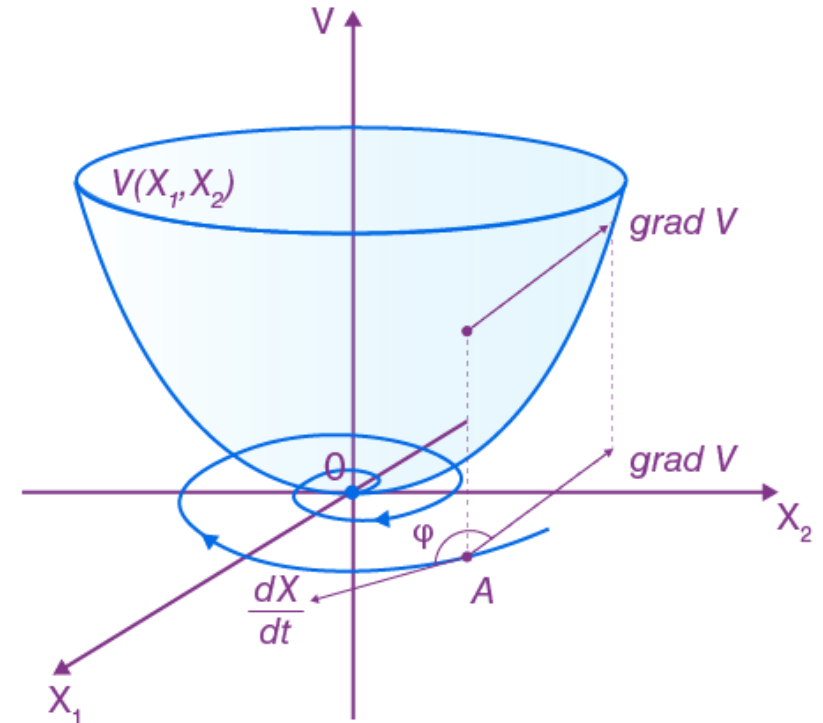


Figure: Example Lyapunov function V shown in 3D space with state dimensions X

Stability Analysis – Cont.

- Neural dynamics introduces some approximation error

- On our neural representation of the Riccati matrix P_n
$$\dot{P}_n = f(P_n) + \epsilon(P_n)$$

- Provide an upper bound for the error

$$\|\epsilon(P_n)\| \leq \bar{\epsilon}$$

- Introduces an additional term on the derivative

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

- Conditions on stability

$$\bar{\epsilon} < \lambda_{\min}(K_n^\top R K_n + Q)$$

Stability Analysis - Cont.

- NEF error arises from two sources

$$\bar{\epsilon} = E_{\text{dist}} + E_{\text{noise}}$$

- E_{dist} : error from finite neural representation
- E_{noise} : error from noise

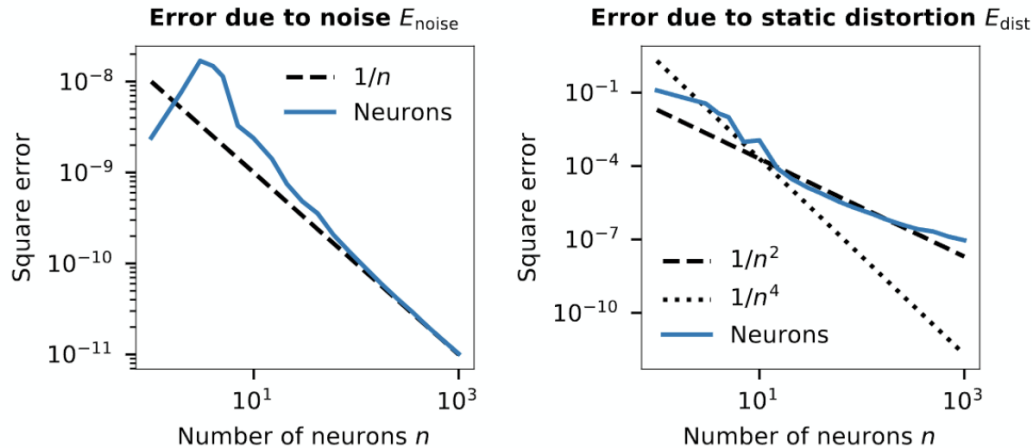


Figure: Error due to noise (left) and error due to state distortion (right) over the number of randomly sampled neurons on a log scale using the NEF.

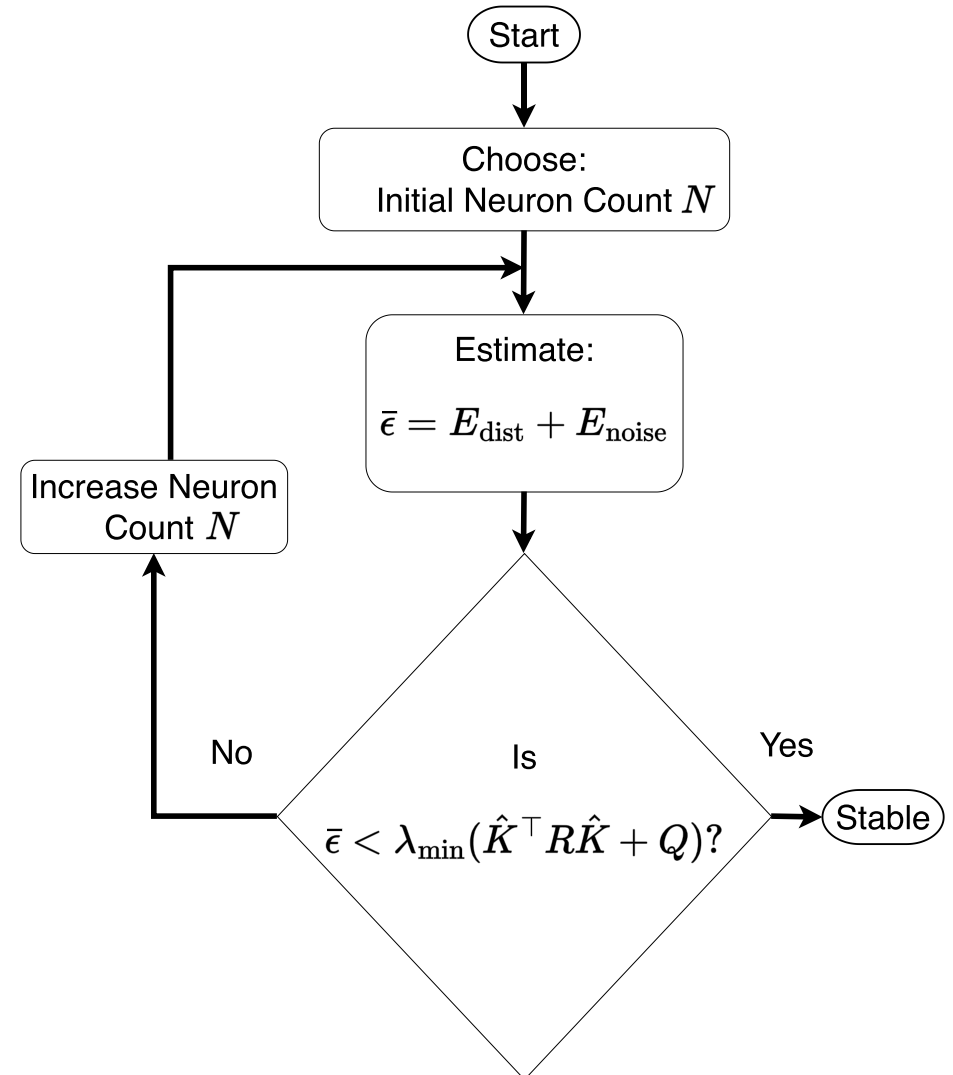


Figure: Design workflow for ensuring closed-loop stability in the proposed spiking LQR controller. The diagram outlines the steps for determining the total neuron count required, evaluating distortion and noise error for stability margins.

Simulation and Results

Experimental Setup

- Quadrotor yaw dynamics for hover
 - Control input: motor voltages

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

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

$$C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad D = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

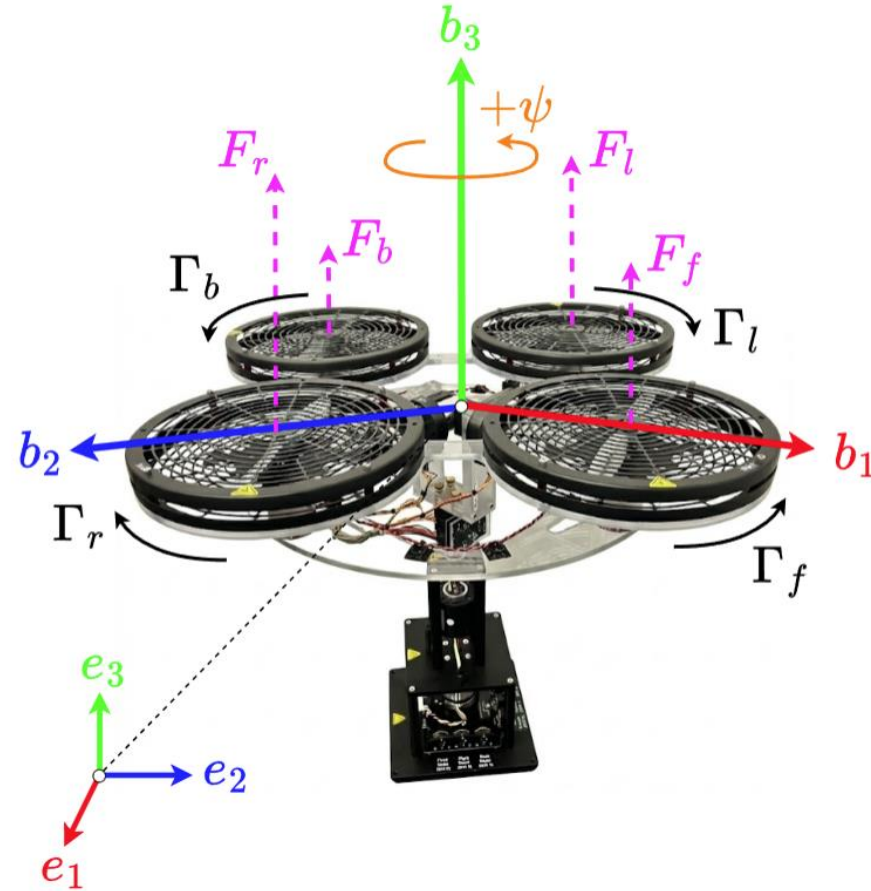


Figure: 3 DoF Hover system diagram. Positive rotations are defined with the right-hand convention

Riccati Dynamics and Convergence

- Evolution of P states over time
- Ideal values are calculated offline for comparison

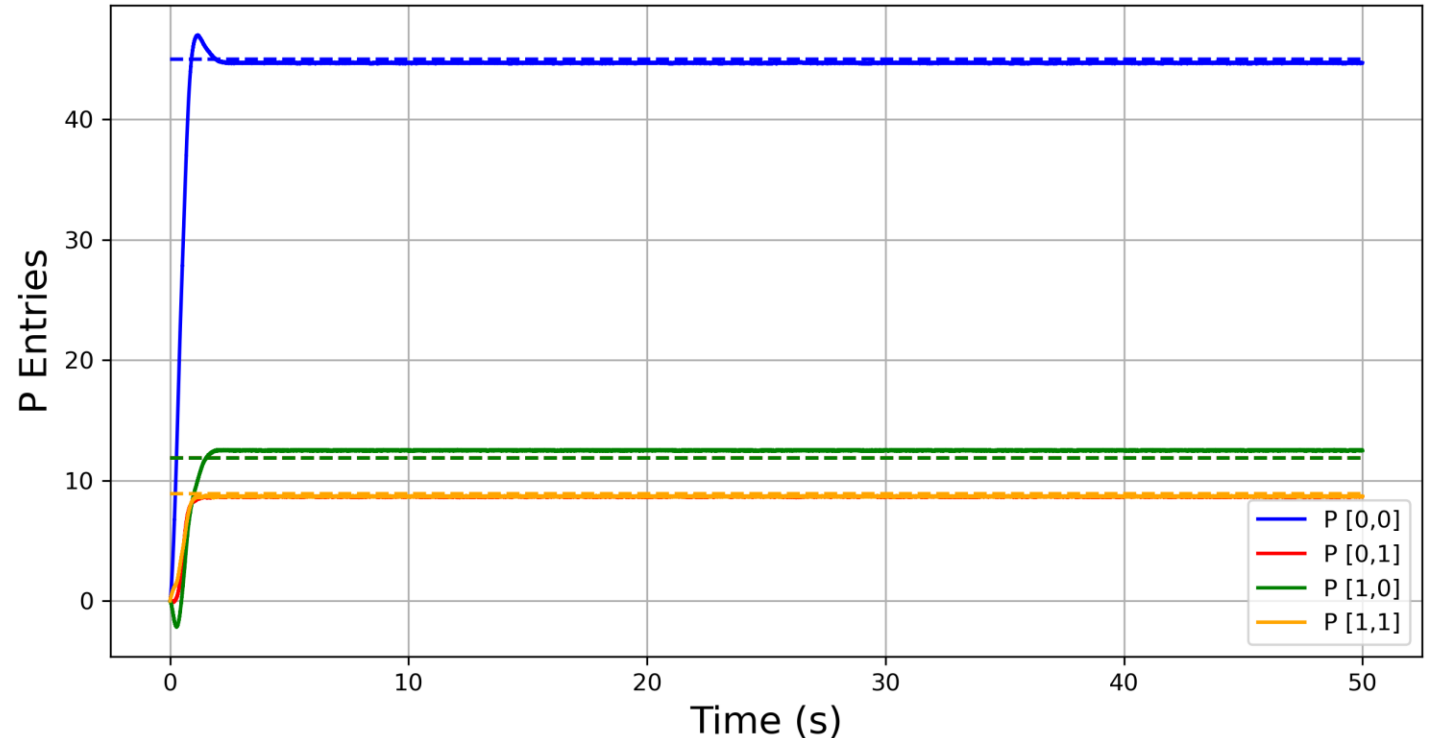


Figure: Time evolution of the entries of the neural Riccati matrix $P(t)$ is represented by solid lines. Dashed lines indicate the optimal steady-state values P .

Tracking Performance

- Tracking results of simulated system
 - Sine wave reference
- Compared against non-spiking, ideal P values for optimal performance

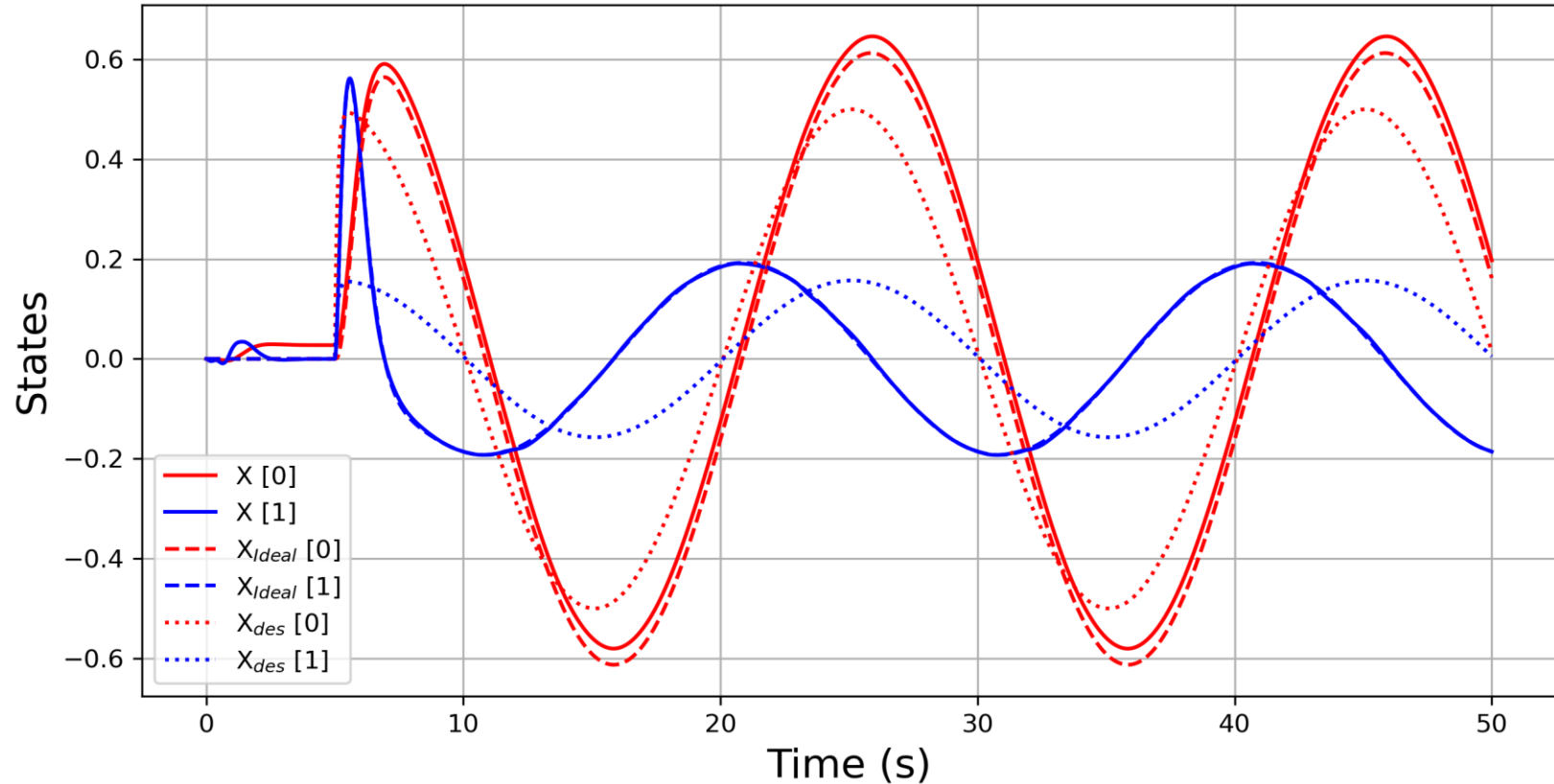


Figure: tracking performance of the state $X(t)$ denoted by the solid lines under neural LQR control. The reference X_{des} is denoted by the dotted lines, and the ideal state X_{Ideal} in the absence of spiking neural networks is denoted by the dashed lines

Stability Validation

- Stability condition

$$\bar{\epsilon} < \lambda_{min}(K_n^T R K_n + Q)$$

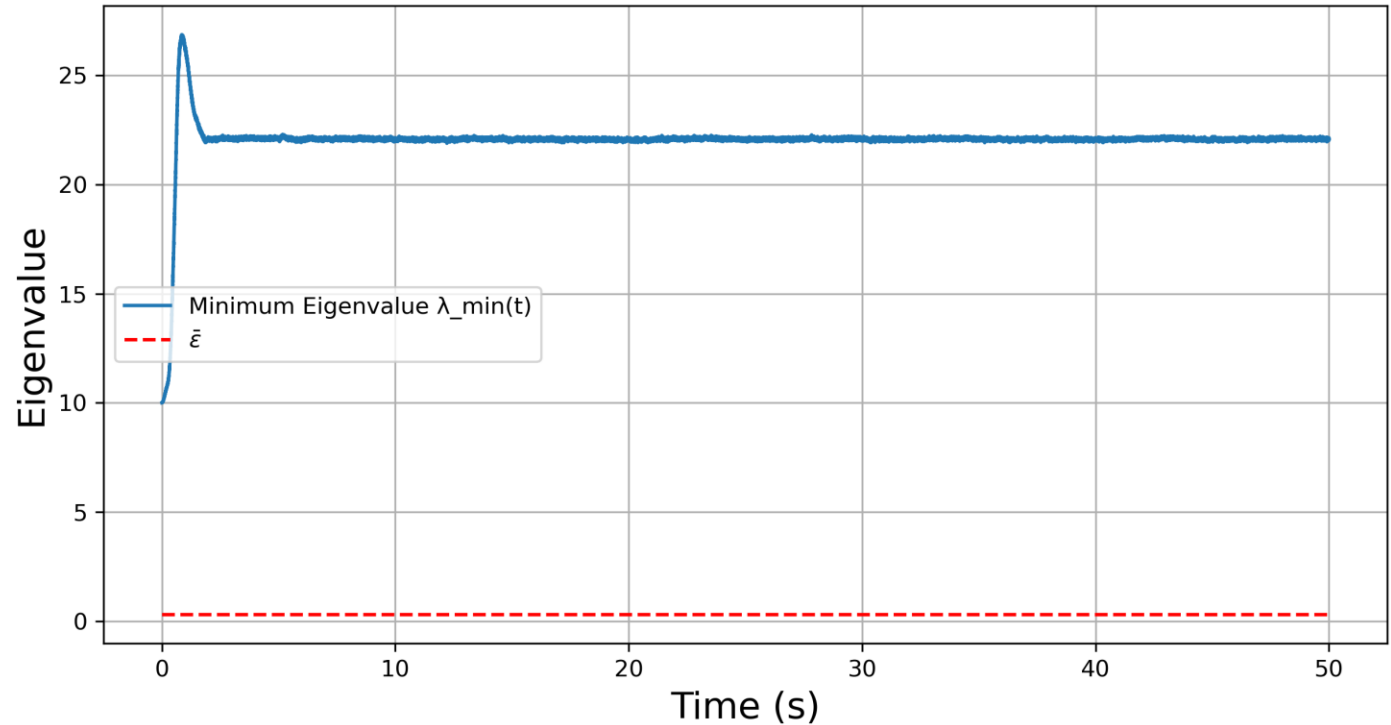


Figure: Minimum eigenvalue $\lambda_{min}(Q + KTRK)$ over time demonstrating stability with respect to the threshold

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

Table: Performance metrics for varying ensemble sizes in both neural ensembles

Discussion and Conclusion

Discussion

- Contributions
 - Dynamically solve the CARE in a spiking neural system
 - Lyapunov based conditions ensuring closed loop stability with spiking solver
 - Demonstration controller, validating system performance and stability
- Stability guarantees with tuning
- Limitations
 - Time-invariant control policy
 - Quadratic scaling of P with respect to state dimension

Future Work

- Physical implementation
 - Actual hover drone
 - Neuromorphic hardware
 - Power analysis
- Representation
 - Representation of P dynamics
- Time varying solutions
 - Leveraging dynamic approach to solving the CARE
 - Adaptive control policies



Nengo Summer School

nengo.ai/summer-school/

A two-week intensive where researchers build cutting-edge brain models and run them on real neuromorphic chips. Now in its 11th year, Nengo Summer School has trained scientists and engineers from around the world.



Real Neuromorphic Hardware

Hands-on time with Braindrop, SpiNNaker & FPGAs — hardware you can't access anywhere else.



World's Largest Brain Model

Learn the framework behind Spaun — spanning perception, memory, and motor control with spiking neurons.



Neural Engineering Framework

Master the NEF: a principled method for compiling cognitive functions into biological neural networks.



ML Meets Neuroscience

Fuse PyTorch, TensorFlow & Keras with spiking architectures for state-of-the-art hybrid models.



Robots & Event Cameras

Deploy neural controllers onto robotic systems and neuromorphic vision sensors in real time.



Your Project, Your Ideas

Bring a research question. Work with global peers and leading faculty for two weeks of intensive collaboration.



Thank You



Mercedes-Benz