Top-down modeling of distributed neural dynamics for motion control

Sruti Mallik[1] and ShiNung Ching[1,2]

Abstract— **In neuroscience a topic of interest pertains to understanding the neural circuit and network mechanisms that enable a range of motor functions, including motion and navigation. While engineers have strong mathematical conceptualizations regarding how these functions can be achieved using control-theoretic frameworks, it is far from clear whether similar strategies are embodied within neural circuits. In this work, we adopt a 'top-down' strategy to postulate how certain nonlinear control strategies might be achieved through the actions of a network of biophysical neurons acting on multiple time-scales. Specifically, we study how neural circuits might interact to learn and execute an optimal strategy for spatial control. Our approach is comprised of an optimal nonlinear control problem where a high-level objective function encapsulates the fundamental requirements of the task at hand. We solve this optimization using an iterative method based on Pontryagin's Maximum Principle. It turns out that the proposed solution methodology can be translated into the dynamics of neural populations that act to produce the optimal solutions in a distributed fashion. Importantly, we are able to provide conditions under which these networks are guaranteed to arrive at an optimal solution. In total, this work provides an iterative optimization framework that confers a novel interpretation regarding how nonlinear control can be achieved in neural circuits.**

I. INTRODUCTION

Optimal control theory has often been applied to understand and emulate biological motion [1]–[3]. In neuroscience, there has been considerable interest in modeling how our brains control a complex range of motions to navigate in a physical environment (i.e., motor control, e.g., [4], [5]). Several insightful questions can be posed in this context. For instance, how do circuits in the brain determine the best strategy for a given task? How does the brain improve its performance over repeated trials or experiences? How do different sub-circuits co-ordinate in perfecting and performing these tasks? It turns out that mathematical models developed using control theoretic frameworks can help us understand these underlying mechanistic questions.

Recently, we proposed a top-down, or 'normative' model of sensory detection by formulating the question as an optimal control problem [6]. In fact, we derived neural circuit architecture and dynamics directly from the optimal solution that turned out to be remarkably biological in nature. Expanding on this idea, here we explore mechanisms by which neurons generating motor commands could learn and implement strategies for spatial control. The basic setup of the problem is: if a dynamical decoder translates neural activity to forces acting along orthogonal axes, then how should neurons in a population activate and deactivate, to induce a target location.

Numerical methods that generate functional models of neural networks have been studied in literature. Of particular relevance to the current work are efforts to construct the dynamics of neurons and synapses to meet specific, high-level control-theoretic objectives [7]–[9]. More specifically, the contributions of this paper are as follows:

- We pose the question of motor control as a continuous time nonlinear optimization problem, and we analyze how the synthesis of a solution can be implemented through synaptic interactions of neurons over multiple timescales.
- We characterize the conditions under which the mathematical framework converges and describe what this signifies for the derived network.

The contents of this paper are organized as follows. In Section 2, we introduce the proposed optimization framework and its solution. In Section 3, we discuss how network dynamics realize the algorithm for solution as well as the solution itself and characterize its convergence. Finally, in Section 4, we demonstrate our framework through a numerical example.

II. OPTIMIZATION FRAMEWORK

A. Problem Formulation

We begin by formally introducing the variables in our problem setup. The variables $\mathbf{p}(t) \in \mathbb{R}^m$ and $\mathbf{v}(t) \in \mathbb{R}^m$ correspond to position and velocity, respectively, of a planar point-mass at time t. Without loss of generality and primarily for ease of description, here we restrict ourselves to $m = 2$. The activity of neural units at time t in given by $\mathbf{x}(t) \in \mathbb{R}^n$.

The motion of the unit point mass is governed by

$$\dot{\mathbf{p}} = \mathbf{C}\mathbf{v} \tag{1}$$

and,

$$\dot{\mathbf{v}} = -\lambda_f \mathbf{v} + \mathbf{b}\mathbf{f}(\mathbf{x}) \tag{2}$$

Here, $\mathbf{C}$ captures the dynamical coupling, if any, existing between motion along the two axes. $\lambda_f > 0$ captures possible dissipation (e.g., due to friction) and the matrix $\mathbf{b} \in \mathbb{R}^{m\times n}$ linearly mixes the contribution of neural units along each dimension of the plane. $\mathbf{f}(\mathbf{x})$ is a non-linear transformation on the activity of neural units. Specifically, $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ is considered to be of the form: $\mathbf{f}(\mathbf{x}) = [f_1(x_1), ..., f_n(x_n)]^T$.

S. Ching holds a Career Award at the Scientific Interface from the Burroughs-Wellcome Fund. This work was partially supported by grants 1653589 and 1724218 from the US National Science Foundation.

[1]All authors are with the Department of Electrical and Systems Engineering, Washington University, St. Louis, MO 63130, USA `sruti.mallik @ wustl.edu`

[2]S. Ching is with the Department of Biomedical Engineering, Washington University, St. Louis, MO 63130, USA `shinung @ wustl.edu`

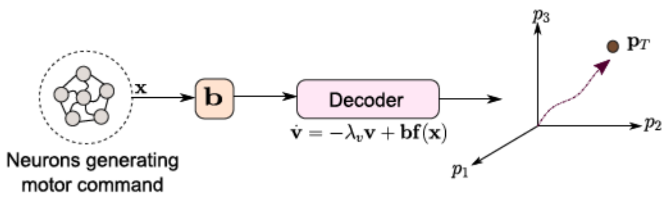
x
b
Decoder
$\dot{\mathbf{v}} = -\lambda_v \mathbf{v} + \mathbf{b}\mathbf{f}(\mathbf{x})$
Neurons generating motor command
p_3
$\mathbf{p}_T$
p_2
p_1

Fig. 1: Neural activity from the brain is decoded into force acting along different axes via a nonlinear dynamical decoder.

Through (2), we posit that the high-dimensional activity of the neural units is decoded into forces acting along axes of motion in a low-dimensional plane(see Figure 1). Neural activity decoded via (2) must enable the system to reach a fixed target position in the plane, i.e., $\mathbf{p}_T$. With this, we specify a fixed-time endpoint objective:

$$\mathbb{J}(\mathbf{x}) = \int_0^T \frac{1}{2}[(\mathbf{p} - \mathbf{p}_T)^T \mathbf{Q}(\mathbf{p} - \mathbf{p}_T) + \overline{\mathbf{x}}^T \mathbf{S}\overline{\mathbf{x}} + \dot{\mathbf{x}}(t)^T \mathbf{R}\dot{\mathbf{x}}(t)]\mathrm{d}t + J_f(T, \mathbf{p}(T), \overline{\mathbf{x}}(T)) \tag{3}$$

Here, $\overline{\mathbf{x}}(t) = \mathbf{x}(t) - \mathbf{x}_b$, where $\mathbf{x}_b$ is the baseline activity of the neural units.

The terms in the integrand can be interpreted as follows: the first term quantifies how far the point mass is from the target location. The second term quantifies the deviation of neural activity from baseline activity, hence providing a measure for energy expenditure. The final term in (3) regularizes rapid fluctuations in neural activity. In addition, there is a penalty for failing to reach the desired location within the given timeframe T. Here, $J_f(T, \mathbf{p}(T), \mathbf{x}(T)) = 0.5[(\mathbf{p}(T) - \mathbf{p}_T)^T \boldsymbol{Q_f}(\mathbf{p}(T) - \mathbf{p}_T) + \overline{\mathbf{x}}^T \mathbf{S}_f \overline{\mathbf{x}}]$. To summarize, the cost function in (3) enforces that the target location be reached within T driven by energy efficient neural activity.

Combining (2) and (3) leads to the problem:

$$\begin{aligned} &\min_{\mathbf{x}} \quad \mathbb{J}(\mathbf{x}) \\ &\text{subject to, } \dot{\mathbf{p}} = \mathbf{C}\mathbf{v} \text{ and } \dot{\mathbf{v}} = -\lambda_f \mathbf{v} + \mathbf{b}\mathbf{f}(\mathbf{x}) \\ &\text{and, } \mathbf{p}(0) = \mathbf{0}, \mathbf{v}(0) = \mathbf{0}, \mathbf{x}(0) = \mathbf{x}_b \end{aligned} \tag{4}$$

B. Solving the Optimal Control Problem

In order solve (4), we reformulate the problem by applying several algebraic transformations to the variables closely following from [6]. Let, $\mathbf{e}(t) = \mathbf{p}(t) - \mathbf{p}_T$ and $\boldsymbol{\omega} = [\mathbf{e}^T, \mathbf{v}^T, \overline{\mathbf{x}}^T]^T$. Note that $\mathbf{x}_b$ is a constant value, therefore $\dot{\mathbf{x}} \equiv \dot{\overline{\mathbf{x}}} = \mathbf{y}$. The optimization problem can now be written as:

$$\begin{aligned} &\min_{\mathbf{y}} \quad \frac{1}{2}\int_0^T [\omega^T \overline{\mathbf{Q}} \omega + \mathbf{y}^T \overline{\mathbf{R}} \mathbf{y}] d\tau + J_f(\omega(T)) \\ &\text{subject to, } \dot{\omega} = g(\omega, \mathbf{y}) \\ &\text{and, } \omega(0) = [\mathbf{0}_m^T, \mathbf{0}_m^T, \mathbf{0}_n^T]^T \end{aligned} \tag{5}$$

Here, $J_f(T, \omega(T)) = \frac{1}{2}[\omega(T)^T \overline{\mathbf{Q}}_f \omega(T)]$, with $\overline{\mathbf{Q}} = diag(\mathbf{Q}, \mathbf{0}_{m\times m}, \mathbf{S})$, $\overline{\mathbf{R}} = \mathbf{R}$ and $\overline{\mathbf{Q}}_f = diag(\overline{\mathbf{Q}}_f, \mathbf{0}_{m\times m}, \overline{\mathbf{S}}_f)$.

With these transformations, (5) becomes finite horizon nonlinear optimization problem. When the constraining system dynamics is linear, this optimization can be easily solved via solution of the Riccati equation [10]. However, in this case due to the nonlinearity $f(\mathbf{x})$, we need to modify our solution method. In the case of our problem, there are two more points of note. First, the neural state space is not low-dimensional (e.g., $n \leq 2$) as such constraints do not occur in nature and secondly, even if we did impose such a constraint, the resulting augmented state vector would still be higher dimensional (as $\boldsymbol{\omega} \in \mathbb{R}^{2m+n}$). Therefore, instead of using Bellman Optimality Principles to find a global solution [11], we opt for local trajectory based methods motivated from Pontryagin's Maximum Principle [12]. Particularly, we solve (5) through an iterative approach adapting from [3].

C. Iterative Algorithm

We begin by positing an initial guess for the optimal neural dynamics, i.e., $\mathbf{y}_0 = \hat{\mathbf{y}}(t)$. Starting from the initial condition given in (5), the nonlinear system $\dot{\boldsymbol{\omega}} = g(\boldsymbol{\omega}, \mathbf{y})$ is forward simulated to yield a trajectory $\boldsymbol{\omega}_0 = \hat{\boldsymbol{\omega}}(t)$. We then consider perturbations $\boldsymbol{\omega}$ and $\mathbf{y}$ to the initial guess [13], such that:

$$\begin{aligned} \boldsymbol{\omega}(t) &= \hat{\boldsymbol{\omega}}(t) + \delta\boldsymbol{\omega}(t) \\ \mathbf{y}(t) &= \hat{\mathbf{y}}(t) + \delta\mathbf{y}(t) \end{aligned} \tag{6}$$

By applying linearization, we deduce:

$$\delta\dot{\boldsymbol{\omega}}(t) = A(t)\delta\boldsymbol{\omega}(t) + B(t)\delta\mathbf{y}(t) \tag{7}$$

where, $A(t) = \frac{\partial g}{\partial \boldsymbol{\omega}}|_\wedge$ and $B(t) = \frac{\partial g}{\partial \mathbf{y}}|_\wedge$ are the Jacobian matrices of g with respect to $\boldsymbol{\omega}$ and $\mathbf{y}$ respectively evaluated at $[\hat{\boldsymbol{\omega}}(t), \hat{\mathbf{y}}(t), t]$. Based on (6) and (7), we can now set up and solve an auxiliary optimization problem in the perturbation variables (see Appendix for details) as follows:

$$\begin{aligned} &\min_{\delta\mathbf{y}} \quad \int_0^T \mathcal{L}_{aux}(\delta\boldsymbol{\omega}(\tau), \delta\mathbf{y}(\tau))d\tau + J_f(\delta\boldsymbol{\omega}(T)) \\ &\text{subject to, } \delta\dot{\boldsymbol{\omega}}(t) = A(t)\delta\boldsymbol{\omega}(t) + B(t)\delta\mathbf{y}(t) \\ &\text{and, } \delta\boldsymbol{\omega}(0) = \mathbf{0}_{n'} \end{aligned} \tag{8}$$

Since both the initially guessed trajectory $\hat{\boldsymbol{\omega}}(t)$ and the perturbed trajectory $\boldsymbol{\omega}(t)$ begin at the same initial condition, $\delta\boldsymbol{\omega}(0) = \mathbf{0}_{n'}$ with $n' = 2m + n$. The Hamiltonian for this auxiliary problem in (8) is given by:

$$\mathcal{H}(t) = \mathcal{L}_{aux} + \boldsymbol{\lambda}(t)^T (A(t)\delta\boldsymbol{\omega}(t) + B(t)\delta\mathbf{y}(t)) \tag{9}$$

Here, $\boldsymbol{\lambda}(t) \in \mathbb{R}^{n'}$ is the costate variable. Now, from the necessary conditions of optimality, $\boldsymbol{\lambda}(t)$ must satisfy the following differential equation:

$$\dot{\boldsymbol{\lambda}}(t) = -\frac{\partial \mathcal{H}}{\partial \delta\boldsymbol{\omega}} \quad \text{with} \quad \boldsymbol{\lambda}(T) = \frac{\partial J_f}{\partial \delta\boldsymbol{\omega}(T)} \tag{10}$$

A stationary condition can be obtained by setting the derivative of the Hamiltonian with respect to $\delta\mathbf{y}(t)$ to zero, such that:

$$\delta\mathbf{y}(t) = -\overline{\mathbf{R}}^{-1} B(t)' \boldsymbol{\lambda}(t) - \hat{\mathbf{y}}(t) \tag{11}$$

Combining (7), (10) and (11), we obtain a Hamiltonian system as follows:

$$\begin{bmatrix} \dot{\delta\omega}(t) \\ \dot{\boldsymbol{\lambda}}(t) \end{bmatrix} = \begin{bmatrix} A(t) & -B(t)\overline{\mathbf{R}}^{-1}B(t)' \\ -\overline{\mathbf{Q}} & -A'(t) \end{bmatrix} \begin{bmatrix} \delta\omega(t) \\ \boldsymbol{\lambda}(t) \end{bmatrix} + \begin{bmatrix} -B(t)\hat{\mathbf{y}}(t) \\ -\overline{\mathbf{Q}}\hat{\omega}(t) \end{bmatrix} \tag{12}$$

with boundary conditions $\delta\omega(0) = \mathbf{0}_{n'}$ and $\boldsymbol{\lambda}(T) = \overline{\mathbf{Q}}(\hat{\omega}(T)+\delta\omega(T))$. Since (12) is a non-homogeneous system, the optimal solution is not a linear state feedback [3], [10]. We assume that the costate variable has the following form:

$$\boldsymbol{\lambda}(t) \equiv \Theta(t)\delta\omega(t) + \theta(t) \tag{13}$$

Using (12), (13) and the boundary conditions, we can write the following coupled differential equations for computing $\Theta(t)$ and $\theta(t)$ backward in time starting from $\Theta(T) = \overline{\mathbf{Q}}_f$ and $\theta(T) = \overline{\mathbf{Q}}_f\hat{\omega}(T)$

$$\begin{aligned} \dot{\Theta}(t) = -\Theta(t)A(t) - A(t)'\Theta(t) + \\ \Theta(t)B(t)\overline{\mathbf{R}}^{-1}B(t)'\Theta(t) - \overline{\mathbf{Q}} \end{aligned} \tag{14}$$

$$\begin{aligned} \dot{\theta}(t) = -[A(t)' - \Theta(t)B(t)\overline{\mathbf{R}}^{-1}B(t)']\theta(t) + \\ \Theta(t)B(t)\hat{\mathbf{y}}(t) - \overline{\mathbf{Q}}\hat{\omega}(t) \end{aligned} \tag{15}$$

Therefore, the solution of the optimization problem in (5) can be found by solving equations (14), (15), backwards in time iteratively and using the solutions in (13) and (11) to update the initial guess as proposed in (6). The iterative process is summarized in Algorithm 1. As the aim of our problem is to reach the target location $\mathbf{p}_T$, we say that the objective is accomplished if $\mathbf{p}(t)$ reaches within $\mathcal{B}_\delta(\mathbf{p}_T)$ with $0 < \delta << 1$. This assumption allows us to introduce a criteria for termination of the iterative procedure.

III. NETWORK DYNAMICS

Solving the optimization problem through the iterative approach yields interesting observations regarding neural dynamics during learning and executing optimal strategies.

A. Dynamics of learning in the network

The iterative procedure begins from an initial guess of control input and corresponding state trajectory, and over successive iterations evolve towards an optimal solution (see Algorithm 1). In neuroscience, this is equivalent to a network attempting to improve its performance over repeated attempts (i.e., a form of iterative learning).

We know that $\dot{\mathbf{x}}(t) = \mathbf{y}(t)$. Combining this with the update equation in Step 6 of Algorithm 1 we have:

$$\dot{\mathbf{x}}_k(t) \leftarrow (1-\eta_k)\dot{\mathbf{x}}_{k-1}(t) + \eta_k\dot{\mathbf{x}}_c(t) \tag{16}$$

In every iteration, the dynamics of the network $\dot{\mathbf{x}}_k(t)$ is therefore a weighted combination of the activity in the previous trial and the corrective dynamics i.e., $\dot{\mathbf{x}}_c(t)$ learned during present trial. In our simulations, we have chosen $\{\eta_k\}_{k\geq 0}$ to be a monotonically decreasing sequence. With this choice, the network increasingly biases itself towards previous experience as the trials progress.

Algorithm 1: Iterative approach for solving Finite Horizon Quadratic Regulator problem constrained by nonlinear dynamics.

```
Initialization $k = 0$, $\omega_0(0)$, $\mathbf{y}_0(t) = \hat{\mathbf{y}}(t)$;
Set *Tol*;
**while** $||\omega_k(t) - \omega_{k-1}(t)|| > Tol$ **do**
    **Step 1**: $k \leftarrow k+1$;
    **Step 2**: (**Forward pass**) Evolve the system
      following $\dot{\omega}_k(t) = g(\omega_{k-1}, \mathbf{y}_{k-1})$ ;
    **Step 3**: Compute $A_k(t)$, $B_k(t)$ for the auxiliary
      problem ;
    **Step 4**: (**Backward pass**) Compute $\Theta_k(t)$, $\theta_k(t)$
      using (14) and (15) backward in time starting
      from $T$;
    **Step 5**: Find the optimal solution of the auxiliary
      problem:= $\delta\mathbf{y}_k(t) =$
      $-\overline{\mathbf{R}}^{-1}B(t)'(\Theta_k(t)\delta\omega_k(t) + \theta_k(t)) - \mathbf{y}_{k-1}(t)$
    **Step 6**: Update $\mathbf{y}_k(t) \leftarrow \mathbf{y}_{k-1}(t) + \eta_k\delta\mathbf{y}_k(t)$;
    **Step 7**: Update step size $\eta_k$, and check criteria
      $||\mathbf{p}_k(t) - \mathbf{p}_T|| < \delta$.
    **Step 8**: Compute $||\omega_k(t) - \omega_{k-1}(t)||$
**end**
```

B. Auxiliary neural population supervises the network

In the iterative method, we derive the optimal perturbation at each iteration that propels the initial guess towards an optimal solution. Now, the auxiliary problem for determining optimal perturbation variables $\delta\omega_k$ and $\delta\mathbf{y}_k$ is a linear optimization problem with a quadratic cost,. This is much more tractable than the original nonlinear optimization problem. We can systematically manipulate the solution to extract the dynamics of an auxiliary population of neurons (see Appendix for details). The purpose of this auxiliary neural population emerging directly from the solution algorithm is to provide feedback to the neurons directly controlling motion(i.e., $\mathbf{x}_k(t)$) based on an evaluation of previous performance. In a sense, $\mathbf{x}^{aux} \in \mathbb{R}^{n_{aux}}$ acts as a 'critic' of the network generating motor command. The dynamics of the auxiliary population can be written as:

$$\begin{aligned} \dot{\mathbf{x}}_k^{aux}(t) = W_k^{\delta e}(t)\delta\mathbf{e}_k(t) + W_k^f(t)\mathbf{x}_k^{aux}(t) + \\ W_k^s(t)\int_0^t e^{-\lambda_f(t-\tau)}F(\mathbf{x}_{k-1})\mathbf{x}_k^{aux}(\tau)d\tau + \\ h(t, \mathbf{x}_{k-1}, \mathbf{e}_{k-1}) \end{aligned} \tag{17}$$

The first term in (17) can be written as $\delta\mathbf{e}_k(t) = \mathbf{p}_k(t) - \mathbf{p}_{k-1}(t)$. This quantifies the anticipated change in position of the system by executing the current strategy. The second term and the third term represents the contribution of fast and slow processing respectively of the activity of these auxiliary neurons. Note that during slow processing, the activity of auxiliary population is gain modulated by $F(\mathbf{x})$ (i.e., rate of change of input to the dynamical decoder with respect to activity of neurons generating motor command). The final term represents the impact of the navigation performance in

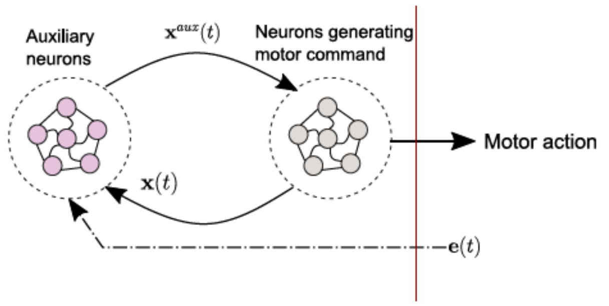

Fig. 2: Improvement in performance occurs through interaction with an auxiliary population. The hyphenated line indicates signals generated at periphery. The vertical red line demarcates the brain regions and the periphery.

the previous trial on the dynamics of the auxiliary population. At each iteration, the auxiliary population receives input from the neurons generating motor command and computes the necessary improvements which is eventually fed back to the command network (see Figure 2). In biological systems, performance in a given task is similarly improved via relevant computations in higher cognitive centers and thereafter feedback to peripheral networks.

C. Convergence of the Iterative Algorithm and its Interpretation

In Sections III.A and III.B, we have shown how the solution of the nonlinear optimization problem (4), can be implemented through distributed neural computations along multiple timescales. In this section, we analyze the convergence of the iterative scheme towards a local minima and interpret it from the perspective of a neural dynamics. In particular, we look at the sequence of state perturbations $\{\delta\omega_k\}_{k\geq 0}$ generated.

Proposition 1: If the nonlinearity $\mathbf{f}(\mathbf{x})$ is doubly differentiable with respect to $\mathbf{x}$, then the sequence of state perturbation $\{\delta\omega_k\}_{k\geq 0}$ converges to a local minimum given that M and N satisfies $\log_{10}\frac{N}{M}\approx 0$ and $0 < M << 1$, where M and N are upper-bounds dependent on specification of system parameters. Under these conditions for all $k \geq K$, the perturbation $\delta\omega_k$ differs from the local minimum $\delta\omega^*$ by no more than $0 < \epsilon << 1$, where $\epsilon = \frac{N}{M}[e^{MT} - 1]$.

Proof: We can write,

$$||\delta\omega_{k+1} - \delta\omega_k|| \leq \int_0^t [\beta_1||A_k - A_{k-1}|| + \beta_2||\Theta_k - \Theta_{k-1}|| + \beta_3||\theta_k - \theta_{k-1}|| + \beta_4||\mathbf{y}_k - \mathbf{y}_{k-1}||]d\tau \tag{18}$$

Note that $E_k(t) = B_k(t)R^{-1}B_k(t)'$. In our formulation, $B_k(t) = \begin{bmatrix} \mathbf{0}_{m\times n} \\ \mathbf{0}_{m\times n} \\ \mathbb{I}_n \end{bmatrix}$ for all k, therefore, $E_k(t) = E$. Let, $\boldsymbol{\xi}_k(t) = ||\delta\omega_{k+1} - \delta\omega_k||$. If the nonlinearity $\mathbf{f}(\mathbf{x})$ is doubly differentiable, then it turns out that:

$$\begin{aligned}\boldsymbol{\xi}_k(t) &\leq \int_0^t (M\boldsymbol{\xi}_{k-1}(\tau) + N)d\tau \\ &= M^k \int_0^t \int_0^{\tau_1} ... \int_0^{\tau_{k-1}} \xi_0(\tau_k)d\tau_k...d\tau_1 + \\ &\quad M^{k-1}N \int_0^t \int_0^{\tau_1} ... \int_0^{\tau_{k-1}} d\tau_k...d\tau_1 + ... + N\int_0^t d\tau_1\end{aligned} \tag{19}$$

where, $M = (\beta_1\alpha_1 + \beta_2\alpha_3 + \beta_3\alpha_5 + \beta_4\alpha_7)$ and $N = (\beta_1\alpha_2 + \beta_2\alpha_4 + \beta_3\alpha_6 + \beta_4\alpha_8)$ (see Appendix for details of the derivation). If $\boldsymbol{\xi}_0(t)$ has an upper bound Γ, then the expression in (19) is bounded by:

$$\begin{aligned}\boldsymbol{\xi}_k(t) &\leq \frac{(MT)^k}{k!}\Gamma + \frac{NM^{k-1}T^k}{k!} + ... + NT \\ &= \frac{(MT)^k}{k!}\Gamma + \frac{N}{M}[\sum_{l=0}^{k}\frac{(MT)^l}{l!} - 1]\end{aligned} \tag{20}$$

As $k \rightarrow \infty$, the first term in (20) goes to zero, while the second term converges to $\frac{N}{M}[e^{MT} - 1]$. Additionally, if $0 < M << 1$, then the second term approaches 0. Therefore, under this situation, the activity auxiliary neurons have converged. Intuitively, when the neural network has converged onto an optimal strategy, it should require very little to no corrective input from the auxiliary population. That is to say, when $||\boldsymbol{\omega}_k||$ reaches an optimal value, $||\delta\boldsymbol{\omega}_k||$ should tend towards zero (see Fig. 3c).

IV. EXAMPLES

Example 1a (Navigating to a target location): We consider a scenario, where the point mass must reach a point $\mathbf{p}_T$

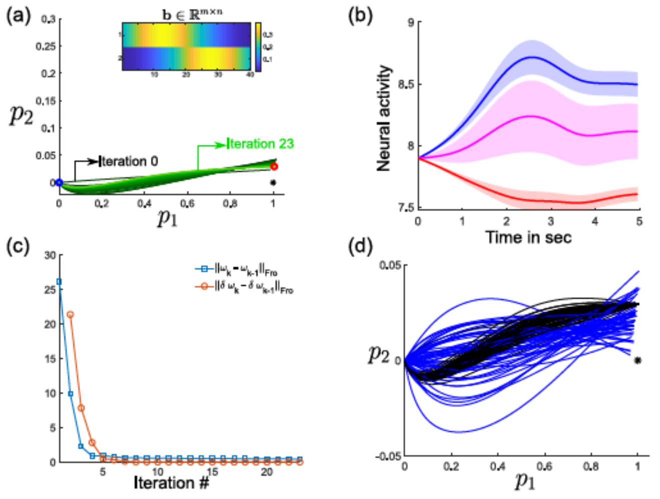

Fig. 3: (a) Optimal trajectory for different iterations. (b) Optimal neural activity upon convergence, the colors arise due to the choice of b. (c) Convergence behavior of the nonlinear problem (in blue) and linear auxiliary problem (in red) (d) Trajectories for random initialization of Algorithm (1) 1: (in blue) trajectory after first iteration, (in black) trajectory after convergence.

within a timeframe of T seconds. In this case, we selected $\mathbf{C} = [0.9, 0.1; 0.1, 0.9]$, $\lambda_f = 0.25$ and the matrix $\mathbf{b}$ such that weights in each row form a Gaussian kernel (see Figure 3a). The non-linearity $\mathbf{f}$ in (2) is chosen to be a sigmoidal and of the form:

$$f_i(x_i) = l + \frac{u - l}{1 + e^{-\alpha(x_i - \beta)}} \text{ for } i = 1,\ldots,n \tag{21}$$

where, u is the upper asymptote and l is the lower asymptote, and α, β are constants that shape the sigmoidal curve.

Finally, we set $\mathbf{Q}$, $\mathbf{S}$, $\mathbf{R}$, $\mathbf{Q}_f$, $\mathbf{S}_f$ as positively scaled identity matrices, such that the resulting $\overline{\mathbf{Q}}$, $\overline{\mathbf{Q}}_f$ are positive semidefinite and $\overline{\mathbf{R}}$ is positive definite and (14) and (15) can be solved [14].

We observe that the algorithm quickly converges to a good candidate solution (see Figure 3a-c). Furthermore, the optimal state perturbation converges over iterations to $\mathbf{0}$. As pointed out in [3], it is important to assess the quality of local minima obtained. For nonlinear optimal control problems (as in Example 1) it is difficult to obtain a global solution. Instead, we investigate the quality of the solution generated by the iterative algorithm using 50 random choices for initialization of $\mathbf{y}(t)$ (see Figure 3d). It turns out that for each of these initializations(in blue) the final trajectory converges to a high fidelity tracking strategy (in black).

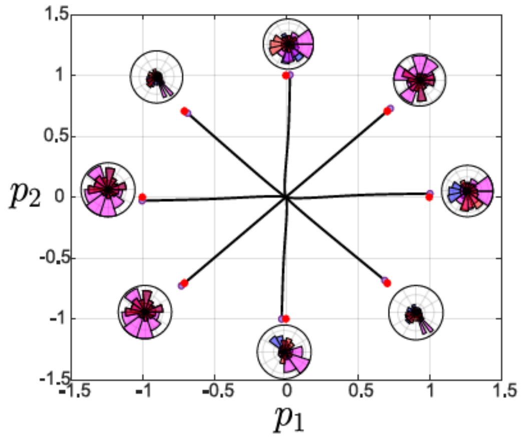

Fig. 4: Optimal trajectory after convergence for 8 target locations (red star) in the two dimensional plane (in black). For each target location neural activity pattern is shown as polar histograms.

Example 1b (Center-out problem): We extend the simulation in 1a further for a standard center-out task [15] and report the resulting neural activity (see Figure 4). We observe that our synthesized network produces unique neural activity patterns for each location, presented here as polar histograms.

V. DISCUSSION AND CONCLUSION

In this work, we present a theory-forward study of how neural networks in the brain could compute efficient strategies for motion control. We specifically derived neural dynamics along multiple timescales that cumulatively result in an optimal solution generated by an iterative algorithm. Essentially, these strategies are embedded within network connectivity. Our results provide hypotheses how neural circuits might coordinate their activity to improve performance over repeated attempts at a given task. Moreover, from an engineering perspective this work can be generalized for implementation of distributed control. Future work would examine the network dynamics obtained for different control tasks and for different formulations of dynamical decoders.

APPENDIX

I. Deriving the cost function for the auxiliary optimization problem

At time t, the cost incurred by the original optimization problem is given as:

$$\mathcal{L}(t, \boldsymbol{\omega}, \mathbf{y}) = [\boldsymbol{\omega}(t)^T \overline{\mathbf{Q}} \boldsymbol{\omega}(t) + \mathbf{y}(t)^T \overline{\mathbf{R}} \mathbf{y}(t)] \tag{22}$$

By Taylor series expansion of (22) around $\hat{\boldsymbol{\omega}}, \hat{\mathbf{y}}$, and by ignoring the higher order terms on the right hand side, we get:

$$\begin{aligned}\mathcal{L}_{aux} = s(t) + \delta\boldsymbol{\omega}(t)^T \mathbf{q}(t) + \delta\mathbf{y}(\mathbf{t})^T \mathbf{r}(t) + \\ \frac{1}{2}\boldsymbol{\omega}(t)^T \overline{\mathbf{Q}} \boldsymbol{\omega}(t) + \frac{1}{2}\mathbf{y}(t)^T \overline{\mathbf{R}} \mathbf{y}(t)\end{aligned} \tag{23}$$

where, $s(t) = \mathcal{L}(t, \hat{\omega}, \hat{\mathbf{y}})$, $\mathbf{q}(t) = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\omega}}|_\wedge$ and $\mathbf{r}(t) = \frac{\partial \mathcal{L}}{\partial \mathbf{y}}|_\wedge$. As $\mathcal{L}_{aux}$ is directly derived from $\mathcal{L}$, it is straightforward to establish that the original nonlinear problem and the auxiliary linear problem are equivalent.

II. Deriving the dynamics of the auxiliary network

The auxiliary optimization problem is solved by computing $\boldsymbol{\Theta}_{\boldsymbol{k}}(t)$ and $\boldsymbol{\theta}_{\boldsymbol{k}}(t)$ in the Backward pass of the Algorithm using (14) and (15). From step 6 of Algorithm 1, we can write:

$$\delta\mathbf{y}_k(t) \equiv \delta\dot{\mathbf{x}}_k(t) = W_k^{\Theta}(t)\delta\boldsymbol{\omega}_k(t) + W_k^{\theta}(t)\theta_k(t) \tag{24}$$

where, $W_k^{\Theta}(t) = -\overline{\mathbf{R}}^{-1} B(t)' \Theta_k(t)$ and $W_k^{\theta}(t) = -\overline{\mathbf{R}}^{-1} B(t)'$. The perturbation in the augmented state vector can be expanded as: $\delta\boldsymbol{\omega}_k(t) = [\delta\mathbf{e}_k(t)^T, \mathbf{v}_k(t)^T, \delta\mathbf{x}_k^T]^T$. We can further write $W_k^{\Theta}(t) = [W_k^{\delta\mathbf{e}}(t) : W_k^{\delta\mathbf{v}}(t) : W_k^{\delta\mathbf{x}}(t)]$. Now, using the linearizations from (7), we can write

$$\begin{aligned}\delta\dot{\mathbf{x}}_k(t) = W_k^{\delta\mathbf{e}}(t)\delta\mathbf{e}_k(t) + W_k^{f}(t)\delta\mathbf{x}_k(t) + \\ W_k^{s}(t)\int_0^t e^{-\lambda_f(t-\tau)} \mathbf{f}_{\overline{\mathbf{x}}}(\mathbf{x}_{k-1})\delta\mathbf{x}_k(\tau)d\tau + \\ W_k^{\theta}(t)\theta_k(t) - \mathbf{y}_{k-1}(t)\end{aligned} \tag{25}$$

Here, $W_k^s(t) = W_k^{\delta\mathbf{v}}(t)\mathbf{b}$, $W_k^f(t) = W_k^{\delta\mathbf{x}}(t)$ and $\mathbf{f}_{\overline{\mathbf{x}}}$ is the Jacobian of the nonlinearity with respect to $\overline{\mathbf{x}}$ evaluated at $\mathbf{x}_{k-1}(t)$. It is straightforward to establish that the last two terms of (25) depend upon activity of the command network (i.e., $\mathbf{x}_{k-1}(t)$), the error generated by execution of the previous strategy (i.e., $\mathbf{e}_{k-1}(t)$) and the parameters that we select for the model. So, we can simplify (25) by writing:

$$\begin{aligned}\delta\dot{\mathbf{x}}_k(t) = W_k^{\delta\mathbf{e}}(t)\delta\mathbf{e}_k(t) + W_k^{f}(t)\delta\mathbf{x}_k(t) + \\ W_k^{s}(t)\int_0^t e^{-\lambda_f(t-\tau)} \mathbf{f}_{\overline{\mathbf{x}}}(\mathbf{x}_{k-1})\delta\mathbf{x}_k(\tau)d\tau + \\ h(t, \mathbf{x}_{k-1}(t), \mathbf{e}_{k-1}(t))\end{aligned} \tag{26}$$

Finally, by setting $\delta \mathbf{x}_k(t) \equiv \mathbf{x}_k^{aux}(t)$ and $F(\mathbf{x}_k) = \mathbf{f}_{\overline{\mathbf{x}}}$ we get (17). Note that, in our setup $n_{aux} = n$. Analysis of network architectures resulting from $n_{aux} \neq n$ remains a scope for future research.

III. Convergence of the iterative algorithm

We look at the sequence of perturbation variable $\{\delta\boldsymbol{\omega_k}\}$ generated by the iterative algorithm [14].

A. β coefficients

$$\frac{d}{dt}[\delta\omega_{k+1} - \delta\omega_k] = [A_k - E\Theta_k]\delta\omega_{k+1} - [A_{k-1} - E\Theta_{k-1}] \\ \delta\omega_k - E\Theta_k - B\mathbf{y}_k + E\Theta_{k-1} + B\mathbf{y}_{k-1} \tag{27}$$

Using the variation of constants formula and by extension of triangle inequality on (27), we get (18). Here, $\beta_1 = ||\Phi_k(t,\tau)||||\delta\omega_k||$, $\beta_2 = ||\Phi_k(t,\tau)||||E||||\delta\omega_k||$, $\beta_3 = ||\Phi_k(t,\tau)||||E||$, $\beta_4 = ||\Phi_k(t,\tau)||||B||$ and $\Phi_k(t,\tau)$ is the transition matrix of the homogeneous system $\frac{d}{dt}(\delta\omega_{k+1} - \delta\omega_k) = [A_k - E\Theta_k](\delta\omega_{k+1} - \delta\omega_k)$.

B. α coefficients

To analyze the upper bound of $||A_k - A_{k-1}||$, we look at $||\frac{\partial \mathbf{f}(\mathbf{x})}{\partial \overline{\mathbf{x}}}|_{\overline{\mathbf{x}}_k} - \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \overline{\mathbf{x}}}|_{\overline{\mathbf{x}}_{k-1}}||$ as $A_k = \begin{bmatrix} \mathbf{0} & \mathbf{C} & \mathbf{0} \\ \mathbf{0} & -\lambda_f \mathbb{I}_m & \mathbf{b}\frac{\partial \mathbf{f}(\mathbf{x})}{\partial \overline{\mathbf{x}}}|_{\mathbf{x}_k} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}$. As $\mathbf{f}(\mathbf{x})$ is doubly differentiable, we can write, $||\frac{\partial \mathbf{f}(\mathbf{x})}{\partial \overline{\mathbf{x}}}|_{\overline{\mathbf{x}}_k} - \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \overline{\mathbf{x}}}|_{\overline{\mathbf{x}}_{k-1}}|| = ||F(\mathbf{x}_k) - F(\mathbf{x}_{k-1})|| \leq L||\overline{\mathbf{x}}_k - \overline{\mathbf{x}}_{k-1}||$ for some $L > 0$. Now, $||\overline{\mathbf{x}}_k - \overline{\mathbf{x}}_{k-1}|| = ||\Gamma||||\omega_k - \omega_{k-1}||$ where $\Gamma = [\mathbf{0}, \mathbf{0}, \mathbb{I}_n]$. Combining the aforementioned expressions and using triangle inequality, we can write:

$$||A_k - A_{k-1}|| \leq \alpha_1 \boldsymbol{\xi}_{k-1}(t) + \alpha_2 \tag{28}$$

where, $\alpha_1 = L||\Gamma||$ and $\alpha_2 = L||\Gamma||||\delta\omega_{k-1}||$.

To obtain the bound on $||\Theta_k - \Theta_{k-1}||$, let us write:

$$\frac{d}{dt}(\Theta_k - \Theta_{k-1}) = -(\Theta_k - \Theta_{k-1})(A_k - E\Theta_k) - \\ (A_{k-1} - E\Theta_{k-1})^T(\Theta_k - \Theta_{k-1}) - \\ (A_k - A_{k-1})'\Theta_k - \Theta_{k-1}(A_k - A_{k-1})$$

By integrating backwards in time, taking the norm and using triangle inequality:

$$||\Theta_k - \Theta_{k-1}|| \leq \int_t^T ||\Phi_k(t,\tau)||(||A_k - A_{k-1}||||\Theta_k|| + \\ ||\Theta_{k-1}||||A_k - A_{k-1}||)||\Phi_{k-1}(t,\tau)||d\tau \\ = \int_t^T \gamma_1 ||A_k - A_{k-1}||d\tau \tag{29}$$

where, $\gamma_1 = ||\Phi_k(t,\tau)||(||\Theta_k|| + ||\Theta_{k-1}||)||\Phi_{k-1}(t,\tau)||$. Now plugging in (28) in (29), we get:

$$||\Theta_k - \Theta_{k-1}|| \leq \alpha_3 \boldsymbol{\xi}_{k-1}(t) + \alpha_4 \tag{30}$$

where, $\alpha_3 = \int_t^T \gamma_1 \alpha_1 d\tau$ and $\alpha_4 = \int_t^T \gamma_1 \alpha_2 d\tau$.

Similarly, we can write:

$$||\theta_k - \theta_{k-1}|| \leq \int_t^T [(\gamma_2\alpha_1 + \gamma_3\alpha_3)\boldsymbol{\xi}_{k-1}(t) + \\ (\gamma_2\alpha_2 + \gamma_3\alpha_4 + \gamma_4)]d\tau \\ = \alpha_5 \boldsymbol{\xi}_{k-1}(t) + \alpha_6 \tag{31}$$

where $\gamma_2 = ||\Phi_k(t,\tau)||||\theta_{k-1}||$, $\gamma_3 = ||\Phi_k(t,\tau)||(||\theta_{k-1}||||E|| + ||B||||\mathbf{y}_{k-1}||)$, $\gamma_4 = ||\Phi_k(t,\tau)||(||\overline{Q}||||\delta\omega_k|| + ||\Theta_k B||||\eta_k||||\delta\mathbf{y}_k||)$, $\alpha_5 = \int_t^T [\gamma_2\alpha_1 + \gamma_3\alpha_3]d\tau$ and $\alpha_6 = \int_t^T [\gamma_2\alpha_2 + \gamma_3\alpha_4 + \gamma_4]d\tau$. Finally,

$$||\mathbf{y}_k - \mathbf{y}_{k-1}|| \leq \alpha_7 \boldsymbol{\xi}_{k-1}(t) + \alpha_8 \tag{32}$$

where, $\alpha_7 = ||\eta_k||||\overline{R}^{-1}B_k^T\Theta_k||$ and $\alpha_8 = ||\eta_k||[||\overline{R}^{-1}B_k^T||(||\Theta_k\delta\omega_{k-1}|| + ||\theta_k||) + ||\mathbf{y}_k||]$. Combining (28), (30), (31) and (32), we get (19).

REFERENCES

[1] Y. Uno, M. Kawato, and R. Suzuki, "Formation and control of optimal trajectory in human multijoint arm movement," *Biological cybernetics*, vol. 61, no. 2, pp. 89–101, 1989.

[2] E. Todorov and M. I. Jordan, "Optimal feedback control as a theory of motor coordination," *Nature neuroscience*, vol. 5, no. 11, pp. 1226–1235, 2002.

[3] W. Li and E. Todorov, "Iterative linear quadratic regulator design for nonlinear biological movement systems." in *ICINCO (1)*, 2004, pp. 222–229.

[4] S. Albrecht, P. Basili, S. Glasauer, M. Leibold, and M. Ulbrich, "Modeling and analysis of human navigation with crossing interferer using inverse optimal control," *IFAC Proceedings Volumes*, vol. 45, no. 2, pp. 475–480, 2012.

[5] S. Bansal, V. Tolani, S. Gupta, J. Malik, and C. Tomlin, "Combining optimal control and learning for visual navigation in novel environments," in *Conference on Robot Learning*. PMLR, 2020, pp. 420–429.

[6] S. Mallik, S. Nizampatnam, A. Nandi, D. Saha, B. Raman, and S. Ching, "Neural circuit dynamics for sensory detection," *Journal of Neuroscience*, vol. 40, no. 17, pp. 3408–3423, 2020.

[7] M. Boerlin, C. K. Machens, and S. Denève, "Predictive coding of dynamical variables in balanced spiking networks," *PLoS Comput Biol*, vol. 9, no. 11, p. e1003258, 2013.

[8] F. Huang and S. Ching, "Dynamical spiking networks for distributed control of nonlinear systems," in *2018 Annual American Control Conference (ACC)*. IEEE, 2018, pp. 1190–1195.

[9] L. F. Abbott, B. DePasquale, and R.-M. Memmesheimer, "Building functional networks of spiking model neurons," *Nature neuroscience*, vol. 19, no. 3, pp. 350–355, 2016.

[10] B. D. Anderson and J. B. Moore, *Optimal control: linear quadratic methods*. Courier Corporation, 2007.

[11] R. Bellman and R. E. Kalaba, *Dynamic programming and modern control theory*. Citeseer, 1965, vol. 81.

[12] R. E. Kopp, "Pontryagin maximum principle," in *Mathematics in Science and Engineering*. Elsevier, 1962, vol. 5, pp. 255–279.

[13] M. Athans and P. L. Falb, *Optimal control: an introduction to the theory and its applications*. Courier Corporation, 2013.

[14] W. Bomela and J.-S. Li, "An iterative method for computing optimal controls for bilinear quadratic tracking problems," in *2016 American Control Conference (ACC)*. IEEE, 2016, pp. 2912–2917.

[15] K. So, K. Ganguly, J. Jimenez, M. C. Gastpar, and J. M. Carmena, "Redundant information encoding in primary motor cortex during natural and prosthetic motor control," *Journal of computational neuroscience*, vol. 32, no. 3, pp. 555–561, 2012.