Learning and encoding motor primitives for limb actions in a brain-like computation approach

Yaoru Sun[1], Haibo Shi[2,*], and Fang Wang[3]

[1] Laboratory of Cognition & Intelligent Computation, Department of Computer Science, Tongji University, Shanghai, 201800, China.
[2] School of Statistics and Management, Shanghai University of Finance and Economics, Shanghai, 200433, China.
[3] Department of Computer Science, Brunel University, Uxbridge UB8 3PH, United Kingdom
[*]Corresponding author: Haibo Shi (shihaibo@sufe.edu.cn).

This work was supported by the Grants from the National Natural Science Foundation of China (91748122), and the Shanghai Science and Technology Committee (17JC1400603).

***Abstract*—Recent neurophysiological studies discovered the sparse rotational patterns in the dynamics of neural population during motor control. In this work, we show that a computational model guided by the dynamical system theory of motor coding can successfully generate the similar network behaviors as found in the electrophysiological studies. The RNN-based model learns the arm reaching control policy from self-generated movements. Essential biomechanical and neural properties including multiphasic neural response and the sparse rotation naturally emerge after training for the movement control tasks. The temporal dynamics in the trained network is analyzed to illustrate how the sparse rotational patterns correlate to the generalization capability of the control policy. We find that the trial-and-error motor learning, which naturally brings in the generalization capability, lead to the existence of low-dimensional manifold in the population dynamics of the motor network.**

***Index Terms*—Motor learning and encoding, motor primitive, motor cortex, brain-like computation, arm action.**

I. Introduction

Motor learning and control are of vital importance to the survival of living organisms. It has been speculated that higher-order cognition of human brain is embedded in the sensori-motor coordination during the interaction between humans and the outside environment [1, 2]. In the past decades, motor learning for encoding of human actions has received intensive studies from fields of both neuroscience and engineering (e.g. robotics and prosthetics) [3, 4]. In particular, the computational modeling provides a qualitatively representative method for testifying various hypotheses for motor learning and control.

A. Background

Recent neuroscience findings have suggested the presence of the dynamical system in the neural population of motor cortices [17]. The evidence supports that the neural activities can be explained with the underlying dynamics of neural population and the motor command can be extracted from the internal dynamics of the motor network, which is opposite to the traditional view that the neural activities in the motor cortices represent the kinetic parameters of the movements (Fig. 1A). Under dynamical system framework, the movement generation is ruled by temporal evolution or state trajectories of combinational activity of neural population (Fig. 1B) [15-20]. The internal dynamics of the motor cortex can be developed autonomously without the sensory stimulation. It thus can functionally form a motor primitive in a manner of central pattern generator (CPG) [21].

Traditional kinematic models [10, 11] have been proposed for learning and producing of movement trajectories to instruct the design of optimal control theory [3, 5, 6] and reinforcement learning [7-9]. These previous models relate the control signal to the specific task parameters, thus in general they do not have the power of reusing and generalizing the movements within the same motor pattern. With movement trajectory changes from time to time, they have to re-plan each trajectory even though these trajectories can be categorized to the same action pattern, in another word, the same motor primitive.

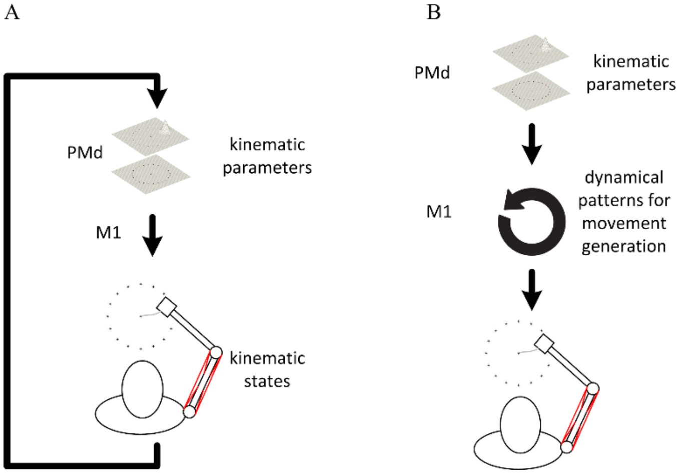

Fig. 1. Two theoretical paradigms for motor generation. A. The representational framework of motor coding. B. The dynamical system framework of motor coding.

In this paper, we propose a neural computation model that incorporates the cortical motor coding mechanisms for robotic motor control. The model exploits a recurrent neural network (RNN) equipped with the neural coding mechanism to implement a transformation from the task information to the time varying motor commands. By examining the neural activities in the model network, the dynamical system hypothesis of neural coding in motor cortices can be verified. In addition, the activities reconstructed in this model can be further analyzed to interpret the origin of the dominant features in the real neural activities.

B. *Related work*

Previous computational models that take RNN dynamics into account for motor generation have captured the behaviors of motor network successfully [29-32]. Studies in [30] demonstrates that the stability-optimized circuits (SOC) could be exploited to illustrate the multiphasic responses of individual neurons in physiological studies. They attributed the multiphasic activity during motor execution as the transient dynamics or selective amplification from the perspective of sensory responses, a feature that was a natural consequence of SOC. Other studies [31, 32] also showed that a RNN trained to generate EMG activities, which were directly recorded from three muscles, displayed both similar activity in both the individual neuron level and rotational patterns at the population level.

However, in the previous studies, the muscle activation signal or end-effector trajectory is required in advance for model training. The models in [31,32] is trained to output the EMG signals of individual muscles in the limited conditions. The low-dimensional structure of internal dynamics is achieved by introducing an important regularization term that penalizes unnecessarily complicated state-space trajectories. On contrary, the motor network in our study is trained with the self-generated movements. The RNN architecture in our model is adapted in the manner of trial and error, i.e., the network parameters are optimized to minimize the control error. So the correlation between the controlling performance and the network dynamics can be further investigated.

There have also been similar methods applying the idea of generating movement trajectory from the goal parameters [34], among which the dynamical movement primitives (DMP) method has been extensively used in motor control studies [12-14]. DMP generates movement trajectories in the manifold of movement space such that the solution space can be constrained within a small number of dimensions. The DMP architecture solves the problem of movement path planning, but leaves the process of producing the motor actuation unsolved. An additional controller module should be used to generate the control commands. This is inconsistent with the neural mechanism of motor control, where the neural activities in the primary motor cortex directly drive the muscle activation [3]. Also, in DMP the dynamical system for planning the movement trajectories is further determined by the evolution of the canonical variable that represents the phase of the movement by the first-order decaying dynamics. Therefore, the canonical variable is updated without the system state [12]. This is not congruent with the motor coding mechanism where the neural state is determined by the internal dynamics of the neural population [15, 16]. More importantly, encoding the motion trajectories of actions means that a learnt neural coding or neural activity pattern in motor cortex from one action mode cannot be re-used for the different trajectories even they can be actually performed using the same motor primitive. Consequently, until now despite of these previous great efforts devoted to the issue of action learning and control, there has been still a big gap between the brain's neural encoding mechanism of motor learning and the computational models to address this issue.

C. *Contributions of current study*

The main contribution of this work can be summarized as follows. First, we construct a computational model for arm reaching that incorporates the recently discovered dynamical system mechanism of motor coding, which learns the control policy from self-generated movements.

Second, using the recently developed jPCA method [16], the low-dimensional manifold in the population activity can be extracted. The learned network exhibits similar features with that of neurophysiological studies, thus provides computational evidence for the dynamical system hypothesis of motor coding.

Third, we further identify the relationship between the control performance and the sparse oscillation in the network activities. It can be testified that the presence of the sparse oscillation is correlated with the capacity of the generalization of the control policy.

Finally, the motor network transforms the spatial information of the kinematic parameters into the time-varying control commands, which could be a promising building block for compound movement control. For example, by switching the high-order command fed into the motor network, the motor network can exert compound controlling commands.

Section II introduces the details of the plant dynamics and the architecture and learning procedure of the controlling network. The training performance as the biomechanical control is validated in section III, before the neural activities are qualitatively compared to the real neural responses in section IV. The sparse rotation is verified and how it contributes to motor control generalization is also shown in section IV. A brief conclusion, significance in related areas of the model and direction for future work are given in section V. Section VI is the appendix which mainly depicts the arm model.

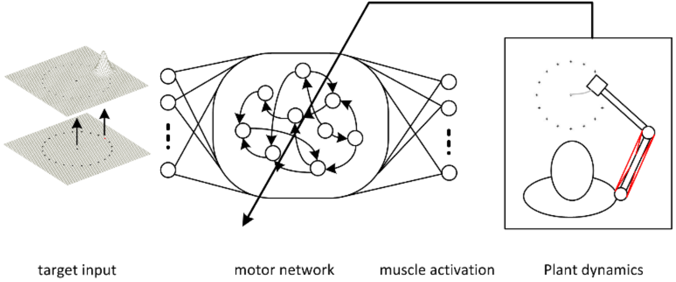

Fig. 2. The causal flow of motor behavior from internal units to the arm plant.

II. NETWORK MODEL

The overall architecture of the neuromusculoskeletal model is depicted in Fig. 2. It models the generation of the arm movement driven by the neural activities. The arm biomechanics is further divided into muscular and skeletal systems. That is, the neural read-out signals drive muscle actuators, and the muscle contraction in turn causes the joint torques for the two-link arm motion.

A. *Architecture and Learning of the internal network*

The internal recurrent neural network serves as a neural controller, which transforms the sensory input and the descending higher-order command into the muscle commands.

This internal network corresponds to the cortical circuits of primary motor cortex (M1) and dorsal premotor areas (PMd) that have direct control of periphery muscle through the cortico-spinal projections. Due to the reciprocal connection in the M1 and PMd areas, we choose the RNN architecture to model the cortical motor circuits. Additionally, RNN allows the internal circling within the network, which can provide the rich internal dynamics suitably for temporal processing. This feature exempts this model from the necessity of external feedback, which is of little interest because the delay is comparable to the movement duration [23].

As shown in Fig. 2, the RNN consists of 100 inter-connected units which receive the sensory input for the start and target positions and the mandatory input for velocity. The command input remains still and neural state converges to a stimulus-specific state by the end of this stage. When the command input is withdrawn, the RNN evolves by its own and projects to 6 output units for 6 muscle activations respectively. In the movement stage, the real-time sensory and proprioceptive feedback is absent [22], and only the internal dynamics of neural population contributes to the control signal. The command input that describes the target position is converted to spatial representation, i.e., points on a grid are activated according to their distance from the goal coordinates. The activation of internal units $r(t)$ is ramped by the $tanh$ function:

$$\mathrm{r(t)} = \mathrm{tanh}\left(\mathrm{W_i} \cdot [\mathrm{s(t)\ I}]^{\mathrm{T}} + \mathrm{W_r} \cdot \mathrm{r(t-1)}\right) \tag{1}$$

The recurrent connections are modified based on the error of motor babblings. Penalty terms and regularities are incorporated in to prevent the neural and muscle activities from growing too much, thus the model is optimized while keeping the recurrent connection and internal activity small. The muscle activity is read out from the internal network with a sigmoid function: $u(t) = \sigma\left(W_o \cdot r(t)\right)$, where W_o is the projection weight, which extracts the output units from internal dynamics. The sigmoid function ensures the positivity of the muscle activation. The muscle activations generated in this network are used to drive the simulated arm model.

The network parameters are tuned to minimize the error between the target state and the final state resulted from the self-generated movement (the inverse arrow from the controlled plant to the network). The input and output weights are fixed while the recurrent weights are learnt. The cost function is the error between the target state $X^*(T)$ and the final state $X(T)$, in addition to the penalty terms related to the muscle $\alpha(t)$ and neural activities $r(t)$ (reflecting the energy cost during movement) :

$$J_c = \frac{1}{2}\sum_{t=1}^{T}\left\{\delta(t-T)\cdot\left(X^*(T)-X(T)\right)^T R^X\left(X^*(T)-X(T)\right) + l_c(R^\alpha\left(\alpha(t)\right), R^r(r(t)))\right\} \tag{2}$$

subject to the forward model of both the network unit activity and the state update:

$$\begin{bmatrix}\dot{X}\\ \dot{r}\\ u\end{bmatrix} = \sigma\left(\begin{bmatrix}M_X(t) & & \\ & [W_i\ W_r] & \\ & & W_o\end{bmatrix}\begin{bmatrix}[X(t) \quad \alpha(t)]^T\\ [r(t) \quad c(t)]^T\\ r(t)\end{bmatrix}\right) \tag{3}$$

where J_c indicates the cost function under c-th condition (goal position, movement duration). Equation (3) gives the integrated form of the updating of arm state, neural state and the muscle activation. $M(t)$ is the linearized approximation of the plant dynamics $\mathrm{f(X(t), u(t))}$ at moment t,where $\mathrm{X(t)}$ and $\mathrm{u(t)}$ are the arm state and muscle input. W_i, W_r, and W_o is the input, recurrent and output weights respectively. The overall cost consists several terms of penalties, where $\left(X^*(T) - X(T)\right)^T R^X\left(X^*(T) - X(T)\right)$ measures the postural deviation of final time point in the intrinsic coordinate frame. $R^\alpha = \left(\alpha(t)^T\alpha(t)\right)$ is the accumulation of muscle activity during the motion execution, that prevents the growing too much. Similar penalty goes with the neural activity for generating the muscle activation signal by incorporating the term $R^r = \left(r(t)^T r(t)\right)$.

We adopt Hessian-free optimization (HF) to tackle the notorious pathological curvature problem in RNN training [24]. HF is a second-order optimization method. The RNN is first unfolded through time and then trained with HF. It is intractable to derive an analytical form of the cost function with respect to the network parameters (the internal weights). Instead, the Gauss-Newton matrix is used as an alternative to the Hessian matrix. The additional benefits are that it can be acquired numerically and is intrinsically positive semi-definite. The Gauss-Newton matrix is derived for every batch of 64 samples and each sample consists of the neural states, dynamics of the plant arm and their gradients with respect to the learning parameters at every moment in the reaching movement. For each batch (64 reaches), the Gauss-Newton matrix is updated and conjugate gradient (CG) is performed accordingly. During the CG iteration, the product of the Gauss-Newton matrix and searching direction vector is obtained via Schraudolph's technique for fast curvature matrix-vector product [25].

B. *Gradients of cost function through trajectory*

The Gauss-Newton matrix is acquired from the gradients of cost function through the motion trajectory, as in Stroeve's method for nonlinear control [26]. During each reaching, the kinematic states of the arm plant are updated forwardly to the final step, and the gradients are derived backwardly.

The arm state $X = \left[\theta_1, \theta_2, \dot{\theta}_1\dot{\theta}_2\right]$ is the concatenation of kinematic and muscle activations states, where θ_i is the i-th joint angle, $\dot{\theta}_\iota$ is the angular velocity of the i-th joint. The gradients of the cost function with respect to network weights at each moment through the whole motion trajectory are derived as following:

$$\begin{cases} \dfrac{\partial J}{\partial X(T)} = \dfrac{\partial l_c(T)}{\partial X(T)} + \left(\dfrac{\partial R(T)}{\partial X(T)}\right)^T \dfrac{\partial J_c(T)}{\partial R(T)} \\ \dfrac{\partial J}{\partial X(k)} = \dfrac{\partial l_c(k)}{\partial X(k)} + \left(\dfrac{\partial R(k)}{\partial X(k)}\right)^T \dfrac{\partial l_c(k)}{\partial R(k)} + \left(\dfrac{\partial X(k+1)}{\partial X(k)}\right)^T \dfrac{\partial J_c}{\partial X(k+1)} \\ \dfrac{\partial J}{\partial \alpha(k)} = \dfrac{\partial l_c(k)}{\partial \alpha(k)} + \left(\dfrac{\partial R(k)}{\partial \alpha(k)}\right)^T \dfrac{\partial l_c(k)}{\partial R(k)} + \left(\dfrac{\partial X(k+1)}{\partial \alpha(k)}\right)^T \dfrac{\partial J_c}{\partial X(k+1)} \end{cases} \tag{4}$$

The elements of matrix $\partial R(k)/\partial X(k)$ and $\partial R(k)/\partial \alpha(k)$ are the partial derivatives of penalty term with respect to arm state the muscle activation respectively. The partial derivatives $\partial X(k+1)/\partial X(k)$ and $\partial X(k+1)/\partial \alpha(k)$ reflect forward dynamics with the Eular integration method:

$$\begin{cases} \frac{\partial X_i(k+1)}{\partial X_j(k)} = \begin{cases} 1+\Delta t\cdot \frac{\partial f_i(X(k+1),\alpha(k))}{\partial X_j(k)} & i=j \\ \Delta t\cdot \frac{\partial f_i(X(k+1),\alpha(k))}{\partial X_j(k)} & i\neq j \end{cases} \\ \frac{\partial X_i(k+1)}{\partial \alpha_j(k)} = \Delta t\cdot \frac{\partial f_i(X(k+1),\alpha(k))}{\partial \alpha_j(k)} \end{cases} \tag{5}$$

Finally the gradient of the cost function with respect to the network weights can be obtained with:

$$\frac{\partial J}{\partial W(k)} = \left(\frac{\partial u(k)}{\partial W(k)}\right)^T \left(\frac{\partial \alpha(k)}{\partial u(k)}\right)^T \frac{\partial J_c}{\partial \alpha(k)} \tag{6}$$

III. PERFORMANCE OF MOTOR CONTROL

As shown in Fig. 3, the motor network successfully learns to generate movements though the internal dynamics. The controlling commands are optimized to solve the inverse dynamics problem of the limb control. During training, the simulated arm is controlled to move its hand to random targets, and the controller network is modified to reduce the motor error.

The RNN controller learnt by HF has a comparable performance with the classic trajectory optimization method such as iterative Linear Quadratic Regulator (iLQR). This is intuitive since both HF and iLQR are based on the second-order optimization. An advantage of the HF-learnt RNN controller over the trajectory optimization-based methods is that it can generalize the control policy to untrained targets.

After being trained for random reaches confined in the working space, the model is instructed to conduct the same center-out reaching tasks as the typical delayed movement generation tasks in the monkey studies [16]. Important biomechanical and neural properties of arm reaching movements have been captured after training. Specifically, it generates relatively straight hand trajectories and bell-shaped velocity profiles (Fig. 3A and Fig. 3B). Velocity profiles are insensitive to the movement amplitude and direction. These features are in consistency with findings of biomechanical experiments in literatures [27]. The analysis is conducted in a 250-ms duration of the movement execution. This time period translates to 25 time steps in the simulation with each time step accounting to a 10-ms bin in the primate experiments.

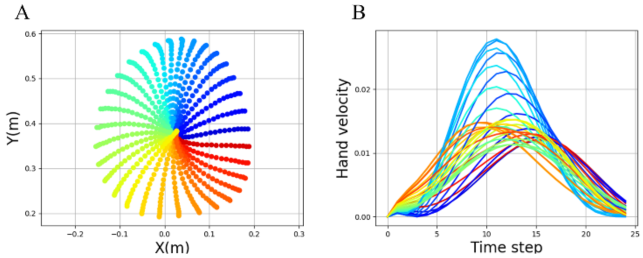

Fig. 3. The performance of the arm control in the center-out task. A. Trajectories of the end effector in the test. D. Velocity profiles of the reaching movements in different directions.

IV. NEURAL RESPONSES AFTER TRAINING

After the validation of the model for generating movements, we examined the neural activation of the internal network. For each control task, the activation of all units is "recorded" for the analysis. The model qualitatively captures the prominent features that are found in the neurobiological studies, specifically, the multiphasic waveforms of individual units, the collective rotation of population activities, and the low-dimension structure or the sparseness of the oscillatory patterns in population dynamics.

A. Neural responses during motor preparation and execution

The activity of motor network is driven by the instruction inputs, and then evolves on its own to generate the motor patterns. We refer the two stages as motor preparation and execution, following the routines in the monkey experiments. During preparation, the simulated arm is maintained at the initial posture. The instruction contains the target position of the reaching movements. The coordinate of the target position is transformed to the activation map of units on a mesh grid, thus the input dimension is increased to the number of grid units, 121. By the end of preparatory stage, the population state of the RNN units converges to certain points in the state space. These stable states in the preparatory phase are, in turn, the initial states of dynamical evolution during the execution phase.

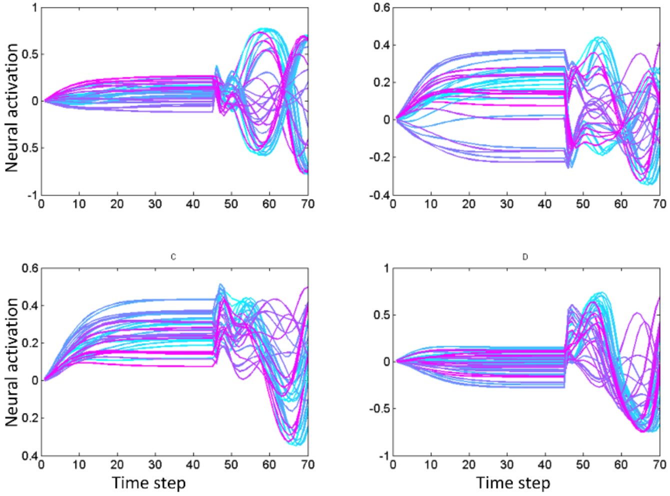

Fig. 4. The multiphasic activation of internal units for 32 reaches.

B. Multiphasic activities of individual neurons

The activities of individual unit across all conditions qualitatively captures the similar dynamics with the waveforms recorded from the M1 neurons. Neural activation shows the similar multiphasic responses during autonomous evolution for different trials in the individual level. The resemblances include the fork-shaped preparation activities and multiphasic oscillations during execution. Starting from zero activity, the activation of individual units exerts divergent trajectories under the stimulation of static input, before asymptotically achieving a stable value respectively (Fig. 4). There is a smooth transition through different orientations, which is crucial to the generalization capability.

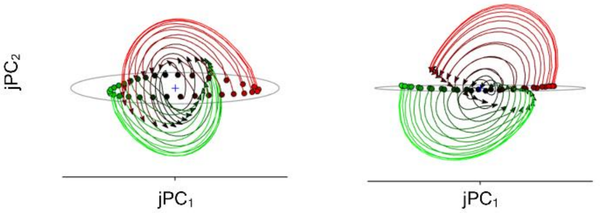

Fig. 5. Rotational trajectories of the internal network for 32 different conditions. The neural activities during movement execution are projected to the plane of the strongest pair of jPCs.

C. *Multiphasic activities of neuron population*

For the population level dynamics, we conduct jPCA on the neural activities of internal network during movement, which has found the consistent rotational structure in collective activities [16] (Fig. 5). The rationale behind this newly developed method is that the temporal processing can be embedded in the network dynamic profiles. The state trajectories in different trials undergo a coherent transformation: a rotation with the same angular velocity. Different kinematic parameters correspond to different initial phases and amplitudes. The neural data recorded from the simulation of the first 6 principle dimensions are prepared for the jPCA analysis, which can be seen as an approximation of the neural activity data with a linear dynamical system, under the constraint of rotational transformation. State rotation is achieved with the skew matrix in linear algebra, thus the analysis reduces to the estimation of the elements of the skew matrix. In current model, the derivative of neural responses $\dot{r}(t)$ is fitted with a skew matrix as $\dot{r}(t) = M_{skew} \cdot r(t)$. The transition matrix is constrained to be skew-symmetric ($M_{skew} = -{M_{skew}}^T$) so that it only has imagery eigenvalues, which correspond to rotational transformation. The search of the dominant state rotation thus reduces to the estimation of the elements of the skew matrix. In our result, the rotation captures 52% of the neural variance of the internal network. The corresponding frequency captured by the first pair of jPCs is 2.4 Hz (with the first pair of eigenvalues 0.140$i$and -0.140i). Also, an unconstrained transition matrix M_{unc} is fitted for comparison: $r(t) = M_{unc} \cdot r(t)$. The similarity between the two transition matrices reflects to what extend the rotational dynamics has contributed to the variance of the overall dynamics. Both the skew matrix and the unconstrained transition matrix are then used to generate the changes of neural state, and contribute to the "recorded" neural date (reserved in 6 principle dimensions).

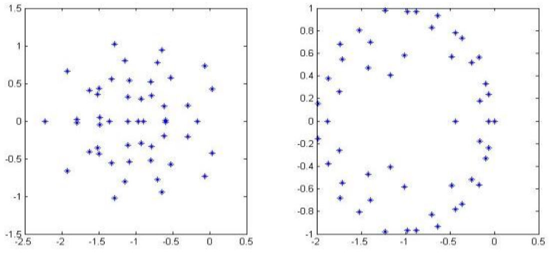

Fig. 6. Eigenvalue analysis of the weight matrix of internal network (left panel) and state transition matrix derived from the population activity (right panel).

D. *Sparseness of the oscillation activities*

As discovered in [16], the variance of neural activity can be accounted by the rotations in first few pairs of jPCs. Only sparse frequencies, which can be discovered by a few strongest rotations that contribute to the variance most, are sufficient for the construction of neural activation (right panel of Fig. 6). We further examined the eigenvalues of the connectivity matrix and the linearly fitted unconstrained matrix, to verify the low-dimensional activity underlying the neural data. The frequencies of these rotations can be also derived from the eigenvalues of the weight matrix W_r of the internal network in Eq. (1), albeit the non-linearity of unit stimulation function (hyperbolic tangent function). The recurrent weight matrix only has only a relatively small number of frequencies which are prominently active, with their real part near zero (left panel of Fig. 6). The imagery part of the eigenvalues is proportional to the frequency of the contributing rotation, while the absolute value of real part implies the decaying rate. The eigenvalues with real part near zero correspond to the sustaining oscillations. The eigenvalues of the unconstrained transition matrix are shown as a comparison with the connectivity matrix in the right panel of Fig. 6.

The sparse frequency oscillation emerges as the optimal solution for the robustness of generating the desired patterns. In other words, a network with delicate and often complicated dynamics, which also reduces the error and achieves the motor tasks, would be sensitive to the perturbations of either unit activity or synaptic strength. The weight adaptation with the cost gradient naturally causes a modification to previously existing oscillations. Due to this continuity, the dramatic change of internal state induced by weight modification during training can be avoided. From this perspective, simple oscillations could survive the plasticity changing, thus provides a strong candidate that can generate locally stable patterns.

E. *Inverse engineering the internal dynamics*

In our model, the continuity of neural activities in motor network across different movements contributes to the generalization capability. Specifically, the gradients comes from the randomly generated motor tasks. The task performance naturally requires the extrapolation of the network dynamics from the already-trained reaching movements. Therefore, the effect of generalization capability in this model can be seen as a regularization.

We conduct an experiment to verify the correlation between the generalization capability and the sparseness of the neural oscillation. The results of the comparison between the generalization performance of the oscillation generators with sparse and disperse frequencies respectively have confirmed that sparsity enhances the generalization capability of control (Fig. 7). The generalization capability is measured by the difference between root-mean-squared-error of testing and training (RMSE). For the sparse internal activity, we use the dominant frequencies found by jPCA, and for the disperse activity we use oscillation generators with frequencies ranging from 0.2 to 10 Hz. The oscillation generators are simulated using the model described in work [28], where rotation matrices are constructed from polar-decomposed components with assigned eigenvalues. Because the eigenvalues are arranged in pairs, the superimposed rotation thus contains 50 frequencies. Composition of sparse oscillations maintains the continuity of internal activities across different tasks. On contrary, from composition of disperse oscillations, only meticulously crafted control policies can be extracted, which cannot be flexibly translated to untrained movements.

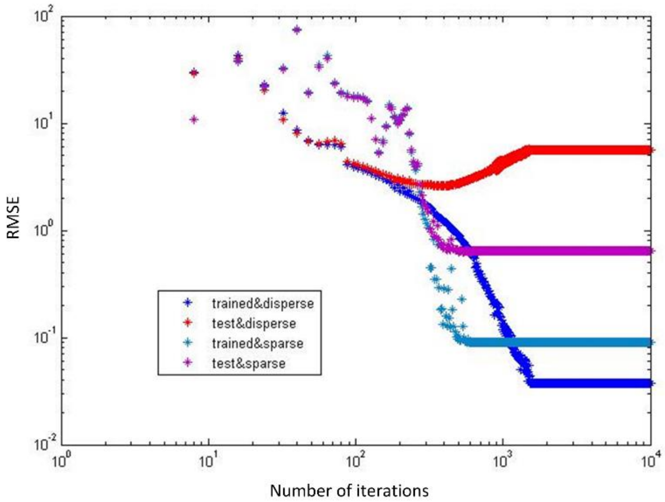

Fig. 7. Generalization capability of the oscillation generators with sparse and disperse dynamics. The generalization capability is measured by the discrepancy between root-mean-squared-error (RMSE) of testing and training. Despite that the trained RMSE of the oscillation generator with disperse rhythms has a more dramatic decay than those with sparse frequencies, the latter one shows lower errors in the test mode.

The regularization for generalization capability on the network learning plays a key role in the sparsification of the network dynamics, and thus can be a supportive factor contributing to the occurrence of low-dimensional manifold in the high-dimensional population activities of the motor network.

The model is inverse-engineered to verify the computational influence of the architecture and learning procedure of the internal network on the simulated arm behavior. We examine the relationship between the changes of network dynamics and structure. We test the filtering effect of the network connectivity on the network activity. A correlation analysis on the unit responses is conducted when they are stimulated by the sine waves of various frequencies. This stimulation is implemented by clamping the activities of some units into sine waves of the same frequency but different phases, by which other units in the network are stimulated to generate collective dynamics. After running this procedure several times over the same connectivity of the network, each with a different frequency, the correlations across internal units can be plotted against the stimulation frequency. As shown by the spectrum patterns of neural response, the internal network only resonates to specific frequencies that are most relevant to the output signals under the trained connectivity. The response shows a frequency preference given a specific connectivity, suggesting that the interplay between internal units under the optimized connectivity generates the most relevant oscillations for the construction of output signal while filtering out the un-tuned ones to sustain the rhythmic activity as locally stable dynamics (Fig. 8)

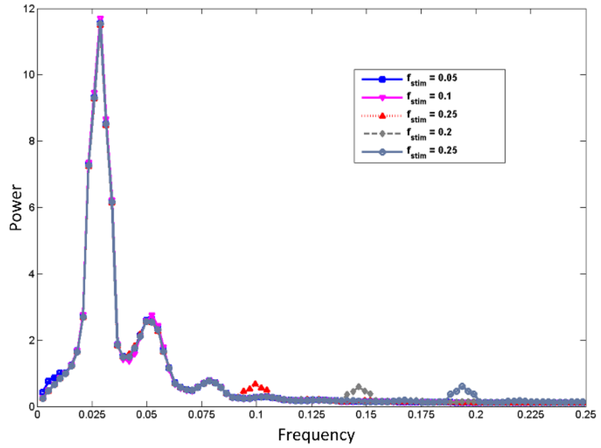

Fig.8 Frequency spectra for network activities of trained connectivity under stimulation of different frequencies.

Furthermore, the factors of the learning procedure to achieve the movement control are also explored. Only a subset of network weights in the architecture is learnt. One tenth of the total are modified during each CG iteration. Obviously, optimization to a specific part of weights can significantly reduce the computing load. And partially tuning the weights, can largely reserve the network dynamics, thus will not severely harm the reaching accuracy. Additionally, adding noise to the network activity helps to improve the control robustness. The network activities that are sensitive to the activity noises are eliminated, while those that are robust to the structural change are remained. The operation achieves similar effect to the regulation terms that penalize weight change to result in dramatic activity change.

The motor network is also trained to eliminate the output during movement preparation. This is biologically necessary, as the EMG signal shows that muscles remain inactivated before movement [16]. Limiting the output ensures that the population state evolves within the null space of the readout weights [18]. Moreover, the suppression on the output activation is, to some extent, transformed to the penalty in the internal unit activity, so that it indirectly poses a stabilizing effect on the network and strengthens the innately decaying tendency of neural activities. This is in agreement with [30] in which stabilization during preparation gives rise to the transient dynamics after the change of the instruction input.

V. DISCUSSION

A. Conclusion

A computational model proposed in this study, which exploits the dynamical system theory of motor coding, successfully reproduced the essential features observed in biological motor control, including multiphasic activities of neural response and the rotational structure of population dynamics. The temporal dynamics in the trained network is analyzed to illustrate how the sparse rotational dynamics correlates to the generalization capability of the control policy. We find that the trial and error learning plays a key role in the occurrence of the low-dimensional manifold of the network dynamics in the high-dimensional population activities.

B. Emergence of oscillatory Activities

Oscillatory activities widely exist in the inter-connected architectures [35]. In this study, it has been shown the rhythmic patterns are not only a product of recurrent connections, but also provide a flexible solution to the control policy of multiple tasks. From this perspective, it is the computational advantage that makes the oscillations as the “pattern generator” of discrete movements. Besides, the dynamics of RNNs need to be robust as it is inevitably exposed to both intrinsic and extrinsic noise. And neural activities are by default sensitive to perturbation of the input stimulation, network states and noises. Sparse rhythmic activities emerge as “locally stable channels” that are robust against heterogeneous perturbation [36]. This is also congruent with the studies in pattern recognition field. For example, the constraint of sparseness improves recognition accuracy of the deep networks [37,38].

C. Serving as motor primitives

With the autonomous dynamics of the internal networks, the initial state specified by the task information can be transformed to temporal patterns of motor signals. This spatio-temporal transformation mechanism can be regarded as the motor primitives for hierarchical control. In this way, the higher-order control module only needs to send out the motor goal in terms of spatial information, leaving the computation of the specific control sequence to the transformation scheme. The temporal evolution of the internal network is contained in the setting of the goal parameters and the initial state [33], in which the initial information can be unfolded to the time-varying commands with deep networks. The motor network can be re-used for different higher commands. Under framework of this study, the complex movements in turn can be generated when a sequence of sub-goals are applied to the neural network where the spatio-temporal motor primitives are accommodated.

D. Future Study

This work has raised some intriguing research directions. First, considering it takes time for the motor network to converge to the initial state for movement execution, directly sequencing the motor primitives would be problematic. Kaufman and colleagues provided an inspiration to tackle this problem, as they found the population dynamics evolving within the null space of the dimension of readout weights [39]. This mechanism allows concatenating motor primitives with no intervention between each other. Second, the current model pertains to model the predictive control (MPC) and controls the external plant using the given physics equations. However, the cortical control relies on a neural representation of the internal model of outside world instead of the precise equations [40]. Hence, it is an appealing line of research to integrate the internal model into this motor generation mechanism under the dynamical system framework.

VI. APPENDIX

Musculoskeletal model

The arm plant is actuated by three pairs of muscle groups. One pair is the biarticular muscle group while the other two respectively actuate one of the shoulder and elbow joints (monoarticular shoulder flexors, monoarticular shoulder extensors, monoarticular elbow flexors, monoarticular elbow extensors, biarticular flexors and biarticular extensors). Each pair of muscle groups is a pair of flexor and extensor. The activation is not directly provided by the network output$u$, but obeys a first-order dynamics:

$$\dot{\alpha}(k) = 1/\tau_d \cdot (u(k) - \alpha(k)) \tag{7}$$

where $\alpha(k)$ is the muscle activation level and $\tau_d = 25ms$ is the first order time constant. Given the muscle activation, the force exerted is also nonlinearly dependent on muscle length and contraction speed:

$$Force(k) = \alpha(k) \cdot f(k) \cdot f(k) \tag{8}$$

Given the muscle tension and the anatomical configuration, which is reflected by the moment arm matrix Γ, the joint torque can be calculated by:

$$Torque(t) = \Gamma \cdot \alpha(k) \cdot f_l(l(k)) \cdot f_v(l(k), v(k)) \tag{9}$$

The moment arm matrix describes the relationship between joint torques and muscle forces under certain gestures. The trivial variations of the moment arms due to gestural change are ignored for simplification, thus Γ is set to a constant matrix:

$$\Gamma = \begin{bmatrix} 2 & -2 & 0 & 0 & 1.5 & -2 \\ 0 & 0 & 2 & -2 & 2 & -1.5 \end{bmatrix}.$$

The columns of Γ represent the individual six muscle groups. The muscle length is fitted using the function of current deviance from the optimal joint angle θ^0 and the optimal length L^0. The matrix θ^0 of size 2×6 indicates the optimal angle of the two joints for each of the six muscle groups. Similarly, six columns in L^0 indicate the optimal length for each of six muscles. Same as in the matrix Γ, zero-value elements in θ^0 and L^0 represent the anatomical absence of the corresponding muscles. For the i-th muscle group, the dependences of current length on the deviance are given as:

$$l_i = 1 + \frac{T_{1,i} \cdot ({\theta^0}_{1,i} - \theta_1)}{{L^0}_i} + \frac{T_{2,i} \cdot ({\theta^0}_{2,i} - \theta_2)}{{L^0}_i} \tag{10}$$

where,

$$\theta^0 = \frac{2\pi}{360} \begin{bmatrix} 15.0 & 5.02 & 0 & 0 & 3.9 & 2.12 \\ 0 & 0 & 80.86 & 109.32 & 92.96 & 91.52 \end{bmatrix}$$
$$L^0 = \begin{bmatrix} 7.32 & 3.26 & 6.4 & 3.26 & 5.95 & 4.06 \end{bmatrix}$$

As the derivative of muscle length $\dot{l}_i$, the muscle contraction velocity can be achieved using a weighted summation of the joint angle velocity $\dot{\theta}_i$, which is also parameterized by moment arm matrix T and the optimal length L^0:

$$\dot{l}_i = \frac{T_{1,i} \cdot \dot{\theta}_1}{{L^0}_i} + \frac{T_{2,i} \cdot \dot{\theta}_2}{{L^0}_i} \tag{11}$$

The muscle length and velocity are normalized by dividing L^0 and thus they can be taken as the relative length and velocity with the unit of L^0 and L^0/s. The nonlinearity terms $f_l(l)$ and $f_v(l(t), \dot{l}(t))$ describe the fascicle force-length relationship and force-velocity relationship. The $f_l(l)$ function is approximated as a 2-sided exponential decay of the deviance of current length from the optimal one to the power of ρ . Differently, the $f_v(l(t), \dot{l}(t))$ function depicts a monotonically decreasing force against velocity:

$$f_l(l) = \exp\left\{-\left(\left|\frac{{l_i}^B - 1}{\omega}\right|\right)^\rho\right\} \tag{12}$$

$$f_v(l_i, \dot{l}_i) = \begin{cases} \frac{V_{max} - \dot{l}_i}{V_{max} + (c_{V0} + c_{V1} l_i)\dot{l}_i} & \dot{l}_i \le 0 \\ \frac{b_V - (a_{V0} + a_{V1} l_i + a_{V2} {l_i}^2)\dot{l}_i}{b_V + \dot{l}_i} & \dot{l}_i > 0 \end{cases} \tag{13}$$

where, $B - 1.55$, $\omega = 0.81$, $\rho = 2.12$, $V_{max} = -7.39$, $c_{V0} = -3.21$, $c_{V1} = 4.17$, $b_V = 0.62$, $a_{V0} = -3.12$, $a_{V1} = 4.21$, $a_{V2} = -2.67$.

The 2-link revolute model of the primate arm has been elaborately described in the literature. Here we adopt the physics formulations given by Todorov [41, 42]. Driven by the joint torque, the plant generates angular accelerations subject to the following equation:

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta}) + B\ddot{\theta} = torque \tag{14}$$

where $\ddot{\theta}$ is the angular acceleration, $M(\theta) \in R^{2\times2}$ is a positive definitesymmetric inertia matrix, $C(\theta, \dot{\theta}) \in R^2$ is a vector reflecting the centripetal and Coriolis forces, $B \in R^{2\times2}$ is the joint friction matrix with respect to the angular velocities, and $torque \in R^2$ is the joint torque. Parameters for these terms are given in the following equations:

$$M(\theta) = \begin{bmatrix} a_1 + 2a_2\cos\theta_2 & a_3 + a_2 cos\theta_2 \\ a_3 + a_2\cos\theta_2 & a_3 \end{bmatrix}$$
$$\mathrm{C} = \begin{bmatrix} -\dot{\theta}_2(2\dot{\theta}_1 + \dot{\theta}_2) \\ \dot{\theta}_1 \end{bmatrix} \mathrm{a}_2\sin\theta_2$$
$$B = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$
$$a_1 = I_1 + I_2 + m_2 {l_1}^2$$
$$a_2 = m_2 l_1 s_2$$
$$a_3 = I_2 \tag{15}$$

where, $b_{11} = b_{22} = 0.05$, $b_{12} = b_{21} = 0.025$, m_i is the mass of the i-th segment (1.4Kg, 1.0Kg), l_i is the length of i-thsegment (33 cm, 30 cm), s_i is the distance from the joint to the mass center of i-th link (11 cm, 16 cm), and I_i is the moment of inertia (0.025 Kgm^2, 0.045 Kgm^2).

The forward dynamics, therefore, can be derived as:

$$\ddot{\theta} = M(\theta)^{-1}(torque - C(\theta, \dot{\theta}) - B\dot{\theta}) \tag{16}$$

The arm state is then updated with the first-order Euler's integration:

$$X(k+1) = X(k) + 1/\tau \cdot f(X(k), torque(k))$$
$$f = \begin{bmatrix} M(\theta)^{-1}(torque - C(\theta, \dot{\theta}) - B\dot{\theta}) \\ \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} \tag{17}$$

where, $M(\theta)$ is the inertia matrix approximated under certain gesture, τ is the time constant, $torque(k)$ is the controlling policy to the arm joints, and $\mathrm{f(X(k), u(k))}$ is the resulting non-linear function of state and torque signal that updates the immediate state of the plant.

REFERENCES

[1] M. R. Gover, "The embodied mind: cognitive science and human experience (Book)," *Mind,* vol. Culture, pp. 295-299, 1996.

[2] V. Gallese and G. Lakoff, "The brain's concepts: The role of the sensory-motor system in reason and language," *Cognitive Neuropsychology,* vol. 22, pp. 455-79, 2005.

[3] E. Todorov, "Direct cortical control of muscle activation in voluntary arm movements: a model." *Nature Neuroscience,* vol. 3, pp. 391-8, 2000.

[4] K. C. Ames, S. I. Ryu and K. V. Shenoy, "Neural dynamics of reaching following incorrect or absent motor preparation," *Neuron,* vol. 81, pp. 438-451, 2014.

[5] E. Todorov and M. I. Jordan, "Optimal feedback control as a theory of motor coordination.," *Nature Neuroscience,* vol. 5, pp. 1226-1235, 2002.

[6] T. M. Tuan, P. Soueres, M. Taix, and E. Guigon, "A principled approach to biological motor control for generating humanoid robot reaching movements," in *IEEE Ras & Embs International Conference on Biomedical Robotics and Biomechatronics*, 2009, pp. 783-788.

[7] J. Morimoto and K. Doya, "Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning," in *Seventeenth International Conference on Machine Learning*, 2000, pp. 623-630.

[8] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *Computer Science,* vol. 8, p. A187, 2015.

[9] J. Morimoto and K. Doya, "Reinforcement learning of dynamic motor sequence: learning to stand up," in *Ieee/rsj International Conference on Intelligent Robots and Systems, 1998. Proceedings*, 1998, pp. 1721-1726 vol.3.

[10] M. L. Latash, J. P. Scholz and G. Schöner, "Toward a new theory of motor synergies," *Motor Control,* vol. 11, pp. 276-308, 2007.

[11] F. Alnajjar, T. Wojtara, H. Kimura, and S. Shimoda, "Muscle synergy space: learning model to create an optimal muscle synergy," *Front Comput Neurosci,* vol. 7, p. 136, 2013.

[12] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: Learning attractor models for motor behaviors," *Neural Computation,* vol. 25, pp.

328-373, 2013.

[13] S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert, "Learning movement primitives," in *Robotics Research, The Eleventh International Symposium, ISRR, October 19-22, 2003, Siena, Italy*, 2003, pp. 561-572.

[14] R. Lioutikov, G. Neumann, G. Maeda, and J. Peters, "Learning movement primitive libraries through probabilistic segmentation," *International Journal of Robotics Research*, vol. 36, pp. 879-894, 2017.

[15] M. S. A. Graziano, "New insights into motor cortex," *Neuron*, vol. 71, p. 387, 2011.

[16] M. M. Churchland, J. P. Cunningham, M. T. Kaufman, J. D. Foster, P. Nuyujukian, S. I. Ryu, and K. V. Shenoy, "Neural population dynamics during reaching," *Nature*, vol. 487, pp. 51-6, 2012.

[17] R. Yuste, J. N. Maclean, J. Smith, and A. Lansner, "The cortex as a central pattern generator," *Nature Reviews Neuroscience*, vol. 6, pp. 477-83, 2005.

[18] K. V. Shenoy, M. Sahani and M. M. Churchland, "Cortical control of arm movements: a dynamical systems perspective," *Annual Review of Neuroscience*, vol. 36, pp. 337-359, 2013.

[19] P. Cisek, "Preparatory activity in premotor and motor cortex reflects the speed of the upcoming reach," *Journal of Neurophysiology*, vol. 96, p. 3130, 2006.

[20] C. Capaday, V. C. Van, C. Ethier, J. Ferkinghoff-Borg, and D. Weber, "Neural mechanism of activity spread in the cat motor cortex and its relation to the intrinsic connectivity," *Journal of Physiology*, vol. 589, pp. 2515-28, 2011.

[21] M. V. Sanchez-Vives and D. A. Mccormick, "Cellular and network mechanisms of rhythmic recurrent activity in neocortex," *Nature Neuroscience*, vol. 3, p. 1027, 2000.

[22] A. Karniel and G. F. Inbar, "A model for learning human reaching-movements," in *Engineering in Medicine and Biology Society, 1996. Bridging Disciplines for Biomedicine. Proceedings of the International Conference of the IEEE*, 1997, pp. 619-620 vol.2.

[23] J. A. Pruszynski, M. Omrani and S. H. Scott, "Goal-dependent modulation of fast feedback responses in primary motor cortex.," *Journal of Neuroscience*, vol. 34, pp. 4608-4617, 2014.

[24] J. Martens and I. Sutskever, "Learning recurrent neural networks with hessian-free optimization," in *International Conference on International Conference on Machine Learning*, 2011, pp. 1033-1040.

[25] N. N. Schraudolph, "Fast curvature matrix-vector products for second-order gradient descent," *Neural Computation*, vol. 14, pp. 1723-1738, 2002.

[26] S. Stroeve, "Neuromuscular control model of the arm including feedback and feedforward components," *Acta Psychologica*, vol. 100, pp. 117-131, 1998.

[27] T. P. Lillicrap and S. H. Scott, "Preference distributions of primary motor cortex neurons reflect control solutions optimized for limb biomechanics.," *Neuron*, vol. 77, pp. 168-179, 2013.

[28] S. Nemati, S. W. Linderman and Z. Chen, "A probabilistic modeling approach for uncovering neural population rotational dynamics," *Working Paper*, vol. 39, p. 123, 2014.

[29] H. Shi, Y. Sun, G. Li and J. Li, "Dynamical Motor Control Learned with Deep Deterministic Policy Gradient", *Computational Intelligence and Neuroscience*, 2018:8535429, 2018.

[30] Hennequin, Guillaume, Vogels, T. Nbsp, Gerstner, and Wulfram, "Optimal control of transient dynamics in balanced networks supports generation of complex movements," *Neuron*, vol. 82, pp. 1394-406, 2014.

[31] D. Sussillo and O. Barak, *Opening the black box: Low-dimensional dynamics in high-dimensional recurrent neural networks*: MIT Press, 2013.

[32] D. Sussillo, M. M. Churchland, M. T. Kaufman, and K. V. Shenoy, "A neural network that finds a naturalistic solution for the production of muscle activity," *Nature Neuroscience*, vol. 18, pp. 1025-1033, 2015.

[33] M. Berniker and K. P. Kording, "Deep networks for motor control functions," *Frontiers in Computational Neuroscience*, vol. 9, p. 32-32, 2015.

[34] W. B. Chen, C. H. Xiong and S. G. Yue, "On configuration trajectory formation in spatiotemporal profile for reproducing human dand reaching movement," *IEEE Transactions on Cybernetics*, vol. 46, pp. 804-816, 2016.

[35] S. Degallier and A. Ijspeert, "Modeling discrete and rhythmic movements through motor primitives: a review," *Biological Cybernetics*, vol. 103, pp. 319-338, 2010.

[36] R. Laje and D. V. Buonomano, "Robust timing and motor patterns by taming chaos in recurrent neural networks," *Nature Neuroscience*, vol. 16, p. 925, 2013.

[37] N. Zeng, H. Zhang, B. Song, W. Liu, Y. Li, and A. Dobaie, "Facial expression recognition via learning deep sparse autoencoders", *Neurocomputing*, 2018: pp. 643-649, 2018.

[38] N. Zeng, Z. Wang, H. Zhang, W. Liu, and F. E. Alsaadi, "Deep Belief Networks for Quantitative Analysis of a Gold Immunochromatographic Strip", *Cognitive Computation*, 8(4): pp. 684-692, 2016.

[39] M. T. Kaufman, M. M. Churchland, S. I. Ryu, and K. V. Shenoy, "Cortical activity in the null space: permitting preparation without movement," *Nature Neuroscience*, vol. 17, pp. 440-8, 2014.

[40] M. Kawato, "Internal models for motor control and trajectory planning.," *Current Opinion in Neurobiology*, vol. 9, pp. 718-727, 1999.

[41] E. Todorov and W. Li, "Optimal control methods suitable for biomechanical systems," in *Engineering in Medicine and Biology Society, 2003. Proceedings of the International Conference of the IEEE*, 2003, pp. 1758-1761 Vol.2.

[42] E. Todorov, "On the role of primary motor cortex in arm movement control," *Progress in Motor Control III*, pp. 125--166, 2003.