<table>
<thead>
<tr>
<th>Title</th>
<th>Self-sustaining rhythmic arm motions using neural oscillators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author(s)</td>
<td>Yang, Woosung; Chong, Nak Young; Kwon, Jaesung; You, Bum Jae</td>
</tr>
<tr>
<td>Citation</td>
<td>IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008. IROS 2008: 3585-3590</td>
</tr>
<tr>
<td>Issue Date</td>
<td>2008-09</td>
</tr>
<tr>
<td>Type</td>
<td>Conference Paper</td>
</tr>
<tr>
<td>URL</td>
<td><a href="http://hdl.handle.net/10119/8490">http://hdl.handle.net/10119/8490</a></td>
</tr>
</tbody>
</table>

Rights

Copyright (C) 2008 IEEE. Reprinted from IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008. IROS 2008, 3585-3590. This material is posted here with permission of the IEEE. Such permission of the IEEE does not in any way imply IEEE endorsement of any of JAIST's products or services. Internal or personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution must be obtained from the IEEE by writing to pubs-permissions@ieee.org. By choosing to view this document, you agree to all provisions of the copyright laws protecting it.
Self-sustaining Rhythmic Arm Motions Using Neural Oscillators

Woosung Yang, Nak Young Chong, Member, IEEE, Jaesung Kwon, and Bum Jae You, Member, IEEE

Abstract—Humans or animals exhibit natural adaptive motions against unexpected disturbances or environment changes. In this paper, we focus on periodic, rhythmic arm motions that can be achieved by using a controller based on neural oscillators. The challenge of this work is to determine appropriate parameters of neural oscillators coupled to a robot arm, accomplishing a given task as well as self-sustaining natural rhythms. For this, an enhanced simulated annealing (SA) algorithm is developed. This work also demonstrates how to technically implement the proposed control scheme to a real robot. Exploiting the entrainment property of neural oscillators coupled to the joints of the arm, we verify that the arm traces a trajectory in such a way that the total energy consumption is minimized, responding to external disturbances.

I. INTRODUCTION

Studies on human-like movement of robot arms have been paid increasing attention, since humans are able to adaptively behave against unknown environment conditions. In particular, human rhythmic movements such as turning a steering wheel, rotating a crank, etc. are self-organized through the interaction of the musculoskeletal system and neural oscillators. In the musculoskeletal system, limb segments connected to each other with tendons are activated like a mechanical spring by neural signals. Thus neural oscillators may offer a reliable and cost efficient solution for rhythmic movement of robot arms. Incorporating a network of neural oscillators, we expect to realize human nervous and musculoskeletal systems in various types of robots.

The mathematical description of a neural oscillator was presented in Matsuoka’s works [1]. He proved that neurons generate the rhythmic patterned output and analyzed the conditions necessary for the steady state oscillations. He also investigated the mutual inhibition networks to control the frequency and pattern [2], but did not include the effect of the feedback on the neural oscillator performance. Employing Matsuoka’s neural oscillator model, Taga et al. investigated the sensory signal from the joint angles of a biped robot as feedback signals [3], [4], showing that neural oscillators made the robot robust to the perturbation through entrainment. This approach was applied later to various locomotion systems [5], [6], [7].

Besides the examples of locomotion, various efforts have been made to strengthen the capability of robots from biological inspiration. Williamson created a humanoid arm motion based on postural primitives. The spring-like joint actuators allowed the arm to safely deal with unexpected collisions sustaining cyclic motions [8]. He also proposed the neuro-mechanical system that was coupled with the neural oscillator for controlling rhythmic arm motions [9]. Arsenio [10] suggested the multiple-input describing function technique to control multivariable systems connected to multiple neural oscillators.

Even though natural adaptive motions were accomplished by the coupling between the arm joints and neural oscillators, the correctness of the desired motion was not guaranteed. Specifically, robot arms are required to exhibit complex behaviors or to trace a trajectory for certain type of tasks, where the substantial difficulty of parameter tuning emerges. The authors have presented encouraging simulation results in controlling the arm trajectory incorporating neural oscillators [11], [12]. This work addresses how to control the trajectory of a real robot arm whose joints are coupled to neural oscillators for a desired task. For achieving this, real-time feedback from sensory information is implemented to exploit the entrainment feature of neural oscillators against unknown disturbances.

In the following section, a neural controller is briefly explained. An optimization procedure is described in Section III to design the parameters of the neural oscillator for a desired task. Details of dynamic responses and simulation and experimental verification of the proposed method are discussed in Section IV and V, respectively. Finally, conclusions are drawn in Section VI.

II. RHYTHMIC MOVEMENT USING A NEURAL OSCILLATOR

We use Matsuoka’s neural oscillator consisting of two simulated neurons arranged in mutual inhibition as shown in Fig. 1. If gains are properly tuned, the system exhibits limit cycle behaviors. Now we propose the control method for dynamic systems that closely interacts with the environment exploiting the natural dynamics of Matsuoka’s oscillator.
where $x_{e(i)}$ is the inner state of the $i$-th neuron which represents the firing rate; $v_{e(i)}$ represents the degree of the adaptation, modulated by the adaptation constant $b$, or self-inhibition effect of the $i$-th neuron; the output of each neuron $y_{e(i)}$ is taken as the positive part of $x_i$, and the output of the whole oscillator as $Y_{out} = \sum_{i=1}^{n} w_{ij} y_{j}$, where $w_{ij}$ is the weight of inhibitory synaptic connection from the $j$-th neuron to the $i$-th neuron, and $w_{ej}$ are also weights from the extensor neuron to the flexor neuron, respectively; $w_{g(i)}$ represents the total input from the neurons inside the network; the input is arranged to excite one neuron and inhibit the other, by applying the positive part to one neuron and the negative part to the other; $T_r$ and $T_s$ are time constants of the inner state and the adaptation effect of the $i$-th neuron, respectively; $s_i$ is the external input, and $g_i$ is the sensory input from the coupled system which is scaled by the gain $k_i$.

Fig. 2 shows two types of mechanical systems connected to the neural oscillator. The desired torque signal to the $i$-th joint can be given by

$$\tau_i = k_i (\theta_i - \dot{\theta}_i) - b_i \dot{\theta}_i,$$

where $k_i$ is the stiffness of the joint, $b_i$ the damping coefficient, $\theta_i$ the joint angle, and $\dot{\theta}_i$ is the output of the neural oscillator that produces rhythmic commands of the $i$-th joint. The neural oscillator follows the sensory signal from the joints, thus the output of the neural oscillator may change corresponding to the sensory input. This is what is called “entrainment” that can be considered as the tracking of sensory feedback signals so that the mechanical system can exhibit adaptive behavior interacting with the environment.

III. OPTIMIZATION OF NEURAL OSCILLATOR PARAMETERS

The neural oscillator is a non-linear system, thus it is generally difficult to analyze the dynamic system when the oscillator is connected to it. Therefore a graphical approach known as the describing function analysis has been proposed earlier [13]. The main idea is to plot the system response in the complex plane and find the intersection points between two Nyquist plots of the dynamic system and the neural oscillator. The intersection points indicate limit cycle solutions. However, even if a rhythmic motion of the dynamic system is generated by the neural oscillator, it is usually difficult to obtain the desired motion required by the task. This is because many oscillator parameters need to be tuned, and different responses occur according to the inter-oscillator network. Hence, we describe below how to determine the parameters of the neural oscillator using the Metropolis method [11], [12] based on simulated annealing (SA) [14], which guarantees convergence to the global extremum [15].

For the process of minimizing some cost function $E, X=[T_r, T_s, w, s, \cdots]^T$ is selected as the parameters of the neural oscillator to be optimized; the initial temperature $T_0$ is the starting parameter; the learning rate $\gamma$ is the step size for $X$. Specifically, the parameters are replaced by a random number $N$ in the range $[-1,1]$ given by

$$X_i = X_{i-1} + \gamma \cdot N.$$

If the change in the cost function $\Delta E$ is less than zero, the new state $X_i$ is accepted and stored at the $i$-th iteration. Otherwise, another state is drawn with the transition probability, $Prob(E)$ given by

$$Prob(E)=(\frac{1}{Z(T)}) \exp(-\frac{\Delta E}{c}) > \gamma,$$

where $\gamma$ is a random value uniformly distributed between 0 and 1. The temperature cooling schedule is $c_i=k_{c}\cdot c_{i-1}$ ($k$ is the Boltzmann constant or effective annealing gain) and $Z(T)$ is a temperature-dependant normalization factor. If $\Delta E$ is positive and $Prob(E)$ is less than $\gamma$ or equal to zero, the new state $X_i$ is rejected. Here the lower cost function value and large difference of $\Delta E$ indicate that $X_i$ is the better solution. If temperature approaches zero, the optimization process terminates.

Even though SA has several potential advantages over
conventional algorithms, it may be faced with a crucial problem. When searching for optimal parameters, it is not known whether the desired task is performed correctly with the selected parameters or not. We therefore added the task completion judgment and cost function comparison steps as shown in Fig. 3 by thick-lined boxes. If the desired task fails, the algorithm reloads previously stored parameters and selects the parameters that give the lowest cost function value. Then the optimization process is restarted with the selected parameters until it finds the parameters of the lowest cost function that allow the task to be done correctly.

IV. CRANK ROTATION OF TWO LINK PLANAR ARM

To validate the proposed control scheme, we evaluate the crank rotation task with a two-link planar arm whose joints are coupled to neural oscillators as shown in Fig. 4. The inter-oscillator network is not established, because the initial condition of the same sign will be equivalent to the excitatory connection between two oscillators. We focus on the entrainment property of the arm.

The crank rotation is modeled by generating kinematic constraints and an appropriate end-effector force. The crank has the moment of inertia $I$ and the viscous friction at the joint connecting the crank and the base. If the arm end-effector position is defined as $(x, y)$ in a Cartesian coordinate system whose origin is at the center of the crank denoted as $(x_0, y_0)$, the coordinates $x$ and $y$ can be expressed as

$$
\begin{bmatrix}
  \dot{x} \\
  \dot{y}
\end{bmatrix} = \begin{bmatrix}
  -r \sin \phi + x_0 \\
  r \cos \phi + y_0
\end{bmatrix} = \begin{bmatrix}
  l_1 c_i + l_2 c_{i2} \\
  l_1 s_i + l_2 s_{i2}
\end{bmatrix}
$$

where $J$ is the Jacobian matrix of $[x, y]^T$. $\phi$ and $\theta_i$ are the crank angle and the $i$-th joint angle, respectively. $l_i$ is the length of the $i$-th link. $c_i, c_{i2}, s_i,$ and $s_{i2}$ denote $\cos \theta_i, \cos(\theta_i + \theta_j), \sin \theta_j$, and $\sin(\theta_i + \theta_j)$, respectively.

By solving Eqs. (7) and (8) simultaneously using Eq. (6), $F$ is obtained as

$$
F = (H(\theta)M(\theta)^{-1}J(\theta)^T + r^2 I^{-1}u(\phi)u(\phi)^T)^{-1}
$$

$$
\cdot (H(\theta)M(\theta)^{-1}(r - V(\phi) + J(\theta, \dot{\theta})\dot{\theta} + r\dot{\phi}(\dot{\phi})^T + CT^{-1}u(\phi)))
$$

Fig. 3 Flowchart of the upgraded SA for task based parameter optimization.

Fig. 4. (a) Schematic robot arm model and (b) real robot arm coupled with the neural oscillator for experimental test.
It is very hard to properly tune parameters of the neural oscillator for attaining the desired rotation task. Moreover, this dynamic model is tightly coupled to crank dynamics as described in Eq. (10). Thus, the proposed parameter tuning approach is divided into the following two steps:

1) Step 1: Find initial parameters of the neural oscillator corresponding to desired inputs of each joint using the cost function given by:

\[
\varphi = \frac{T - T_c}{T_c} + v \cdot \max\left(0, \frac{A_d - C}{B} - 1, 0\right) \tag{10}
\]

subject to

i) \( A_{\text{min}} \leq A_d \leq A_{\text{max}} \)

ii) \( |A_d - C| \leq B \)

where \( C = (A_{\text{max}} + A_{\text{min}})/2 \), \( B = (A_{\text{max}} - A_{\text{min}})/2 \); \( A_d \) is the desired amplitude of the neural oscillator for the rotation task, \( A_{\text{max}} \) and \( A_{\text{min}} \) are the maximum and minimum amplitude constraints, respectively; \( T \) and \( T_c \) denote the desired and measured natural frequencies of the output generated by the neural oscillator, respectively. \( v \) is the performance gain.

2) Step 2: Using the initial parameters obtained by Step 1, run the proposed SA algorithm as illustrated in Fig. 3. The cost function for the crank rotation includes the velocity of the rotation, torque, and consumed energy.

Implementing Step 1 and Step 2 in sequence, we are able to acquire the appropriate initial and tuned parameters as seen in Table II. Figure 5 (a) indicates a cooling state in terms of cooling schedule. Cooling or annealing gain \( K \) is set as 0.95. It can be observed in Fig. 5 (b) that the optimal process was well operated and a better solution at the lowest cost function was obtained iteratively. As expected, when the tuned parameters are employed to perform the given task, a stable motion could be accomplished as shown in Fig. 5. It is evident in Fig. 5 (c) that initial transient responses disappear due to the entrainment property of the neural oscillator. This property enables the arm to sustain the given task against changes in parameters of arm kinematics and dynamics as well as disturbances.

<table>
<thead>
<tr>
<th>TABLE II</th>
</tr>
</thead>
<tbody>
<tr>
<td>INITIAL AND TUNED PARAMETERS OF THE NEURAL OSCILLATOR WITH ROBOT ARM MODEL</td>
</tr>
<tr>
<td><strong>Initial parameters</strong></td>
</tr>
<tr>
<td>Inhibitory weight ((w))</td>
</tr>
<tr>
<td>Time constant ((T_r))</td>
</tr>
<tr>
<td>((T_a))</td>
</tr>
<tr>
<td>Sensory gain ((k))</td>
</tr>
<tr>
<td>Tonic input ((s))</td>
</tr>
</tbody>
</table>

**Robot Arm Model**

| Mass 1 \((m_1)\), Mass 2 \((m_2)\) | 2.347kg, 0.834kg |
| Inertia 1 \((I_1)\), Inertia 2 \((I_2)\) | 0.0098kgm\(^2\), 0.0035kgm\(^2\) |
| Length 1 \((\ell_1)\), Length 2 \((\ell_2)\) | 0.224m, 0.225m |

Fig. 5. (a) Temperature transition for cooling schedule, (b) A transition of total cost function level, (c) The end-effector trajectory of two-link arm (d) The output of joint angle. The red dash line is the first joint angle and the second joint angle is drawn by the blue thin line

V. EXPERIMENTS WITH A REAL ROBOT ARM

To validate the proposed control scheme described in Section IV, we employed a real robot arm with 6 degrees of
freedom (see Fig. 4 (b)) and constructed a real time control system. This arm controller runs at 200 Hz and is connected via IEEE 1394 for data transmission at 4 kHz. ATI industrial automation’s Mini40 sensor was fitted to the wrist joint of the arm to detect external disturbances. The optimized parameters in Table II were used for the neural oscillator.

Fig. 6 shows the arm kinematics. Since the crank motion is generated in the horizontal plane, $q_3$ and $q_4$ are set to 90°. The initial values of $q_3$ and $q_6$ are set to 0°, respectively. $q_2$ and $q_3$, corresponding to $\theta_1$ and $\theta_2$ in Fig. 4 (a), respectively, are controlled by the neural oscillators and the constraint force given in Eq. (10). The constraint force enables the end-effector to trace the outline of the (virtual) crank. Hence, the end-effector can draw the circles as shown in Fig. 7 (see the overlapping circles in the center part of the figure).

Now, we will examine what happens in the arm motion if additive external disturbances exist. Arbitrary forces are applied to the end-effector at 15s, 28s, 44s, 57s, 73s and 89s sequentially as shown in Fig. 8. We first pushed the end-effector along the minus x direction. The force sensor value in the x and y direction are added to Eq. (10). Then, the joint angles change according to the direction of the applied force, which makes the neural oscillators entrain the joint angles as shown in Fig. 9. The solid line is the output of the neural oscillator connected to the first joint ($q_2$) and the dashed line indicates that of the neural oscillator connected to the second one ($q_3$). Hence a change in the output of the neural oscillator causes a change in the joint torque. Finally the joint angles are modified as shown in Fig. 10, where the bottom plot is the output of $q_2$ and the top one is the output of $q_3$. Fig. 11 shows the snap shots of the simulated crank motion by the robot arm, where we can observe that the end-effector traces the circle well, and adapts its motion when an external force is applied to it.

Table III compares the power consumption of the robot arm performing the above task with different parameters of the neural oscillator. The parameters were drawn arbitrary among the ones that guarantee a successful completion of the task. If the optimized parameters (set D) were employed, the most energy-efficient motion was realized.

<table>
<thead>
<tr>
<th>TABLE III</th>
<th>POWER CONSUMPTION ACCORDING TO THE SELECTED PARAMETER SET OF THE NEURAL OSCILLATOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
<td>Parameter set</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------</td>
</tr>
<tr>
<td>Inhibitory weight $(w)$</td>
<td>2.0</td>
</tr>
<tr>
<td>Time</td>
<td>0.25</td>
</tr>
<tr>
<td>Sensory gain $(k)$</td>
<td>0.5</td>
</tr>
<tr>
<td>Tonic input $(s)$</td>
<td>1.0</td>
</tr>
<tr>
<td>Measured current [A]</td>
<td>60.0</td>
</tr>
<tr>
<td>Power [W] Consumption</td>
<td>89.808</td>
</tr>
</tbody>
</table>

Fig. 6. Kinematic parameters of the robot arm

Fig. 7. The trajectory drawn by the end-effector of the arm

Fig. 8. The external forces measured by the force sensor in the x and y direction
VI. CONCLUSION

We have presented an example of human-like behavior of a planar robot arm whose joints were coupled to neural oscillators. In contrast to existing works that were only capable of rhythmic pattern generation, our approach allowed the robot arm to trace a trajectory correctly through entrainment. For achieving this, we proposed an optimization approach for obtaining the parameters of the neural oscillator modifying the simulated annealing method. Simulation and experimental results showed the effectiveness of the proposed approach. Moreover, it was demonstrated that the robot arm could adaptively behave responding to external disturbances keeping the shape of the trajectory unchanged. This approach will be extended to a more complex behavior toward the realization of biologically inspired robot control architectures.

REFERENCES