A Bio-inspired Mechanism for Learning from Mirrored Demonstrations

Omar Zahra, Silvia Tolu, Peng Zhou, Anqing Duan, and David Navarro-Alarcon

The Hong Kong Polytechnic University, Department of Mechanical Engineering, KLN, Hong Kong
Technical University of Denmark, Department of Electrical Engineering, Copenhagen, Denmark

Correspondence*: Omar Zahra
omar.zahra@connect.polyu.hk

ABSTRACT

Different learning modes and mechanisms allow faster and better acquisition of skills as widely studied in humans and many animals. Specific neurons, called mirror neurons, are activated in the same way whether an action is performed or simply observed. This suggests that observing others performing movements allows to reinforce our motor abilities. This implies the presence of a biological mechanism that allows creating models of others’ movements and linking them to the self-model for achieving mirroring. Inspired by such ability, we propose to build a map of movements executed by a teaching agent and mirroring the agent’s state to the robot’s configuration space. Hence, in this study, a neural network is proposed to integrate a motor cortex-like differential map transforming motor plans from task-space to joint-space motor commands and a static map correlating joint-spaces of the robot and a teaching agent. The differential map is developed based on spiking neural networks while the static map is built as a self-organising map. The developed neural network allows the robot to mirror the actions performed by a human teaching agent to its own joint-space and the reaching skill is refined by the complementary examples provided. Hence, experiments are conducted to quantify the improvement achieved thanks to the proposed learning approach and control scheme.

Keywords: Robotics, spiking neural networks, sensor-based control, visual servoing, imitation learning.

INTRODUCTION

Robots are involved nowadays in many demanding and challenging tasks. With the aim to keep up with the pace of such demands, adaptability and novel learning techniques are essential in robots. One of the biologically inspired methods for learning is learning by demonstration or imitation, where the robot is taught by a teaching agent to execute a specific task. An issue that arises is relating the Cartesian space of both the teaching and the robot required for direct teaching from demonstrations (Argall et al., 2009; Ravichandar et al., 2020). In primates, specific neurons in several brain regions, called mirror neurons, are proven to trigger almost the same output while executing or observing the same task (Heyes, 2010; Cook et al., 2014). Consequently, these neurons are considered a key component in learning and refining motor skills in primates (Oztop et al., 2006; Jacoboni, 2009). A biologically inspired mechanism is introduced in
this study to functionally replicate the ability to learn through demonstrations. However, unlike other works in which the robot was required to just copy a certain motor skill, this work aims for the improvement of an acquired/learnt skill (i.e., target reaching) through imitation. In (Iacoboni and Mazziotta, 2007), mirror neurons would respond to intended tasks even with occlusions occurring indicating the sensitivity of these neurons to specific skills/actions rather than joints movements. Hence, most studies focus on monitoring the mirroring activity in the high-order brain regions as these regions are responsible for motion planning. A study in monkeys investigated the activity in the primary motor cortex, responsible for motion transformation, after learning step-tracking while performing and observing the task (Dushanova and Donoghue, 2010). A wide set of neurons was found to attain activity while observing similar to that during acting while preserving the same preferred direction of activity, only with less amplitude. This occurs only while observing a task that was already learnt by the monkey. This indicates that mirror neurons exist even in lower-order regions and may contribute to the refinement of the learnt skills.

Consider a system that builds a map without any prior knowledge about body kinematics, analogous to formation of a transformation map in the motor cortex for new born babies (Zahra et al., 2020b). Through motor babbling, a training data set is generated to allow building the desired map correlating the body state and the motor commands required to produce an intended motion. However, since no inverse kinematic solver or initial model of the kinematic relations is present, the motor babbling commands correspond to random movements in joint-space. Such movements are waving-like motions for the nature of movement for mainly revolute joints utilized in the studied case. It was observed that the error in the reaching actions is highly related to the collected training data of waving-like motion. This was concluded from the longer time and higher deviation from the straight target path to the target point. Hence, an auxiliary teaching mechanism is proposed to enrich the training data. One solution proposed in (Kormushev et al., 2015), where a kinematic-free scheme for robot control was proposed based on generating exploratory motions to find proper motor actions. In this study a more directed data collection is proposed where the candidate mechanism relies on learning by imitating a human agent providing more direct teaching examples. Such examples make up for the lack of proper joint coordination during motor babbling to produce motion in a straight path between numerous points in the task-space.

Surveys of different systems developed for learning from demonstrations discuss the different learning modes and challenges faced by each mode (Argall et al., 2009; Ravichandar et al., 2020). The studied case involves learning from external observations, where demonstrations are performed by a teaching agent with no sensors attached on the agent. Additionally, the policy to be learned in this case aims for low-level control of the robot in the joint space. As this case involves passive observation imitation learning, it suffers from the correspondence issue to transform the demonstration from the teacher’s joint space to the robot’s joint space. In (Shavit et al., 2018), a dynamical system (DS) is proposed to learn from kinesthetic demonstrations. The DS is then capable of computing the desired motion to be executed in joint space to reach a target in task-space. However, no mechanism for learning from demonstrations of a teaching agent is included in the study as teaching occurs only by moving the robot links manually to execute the task (i.e., kinesthetic learning only). In (Tieck et al., 2017), a spiking neural network (SNN) is introduced to reproduce the grasping motion of a hand. The data collected during a human hand grasping different objects is recorded to train the network. Then, the SNN guides the fingers of robotic hand to grasp the objects. While the SNN reproduces the pattern of recorded movements, it does not address the case where different link lengths exist in the teaching agent/hand and the robot. Moreover, the error recorded for the joints is relatively big at the end of the training.
In this study, an SNN is developed to guide the motion of a robot through joint space motor commands in a visual servoing task. Without any prior knowledge about the robot configuration and intended direction of motion, the SNN is trained through motor babbling to provide adequate motor commands. The developed sensorimotor map is then refined by imitating the movements of a teaching agent, a human arm movement in this study, to make up for the missing knowledge about the desired movements. The teaching examples are transformed into robot coordinates through a network developed based on the self-organizing map (SOM) and Hebbian learning plasticity rule. Hence, this study contributes to:

- Solving the correspondence issue via SOMs and biologically inspired plasticity rule
- Improving the performance of a feedforward SNN \cite{Zahra2020b} relying on Bayesian optimization and inhibitory interconnections
- Validating the improvement in representation capabilities of the developed SNN via complementing the training data

For the best of our knowledge, this is the first study to utilize SOMs to solve the correspondence issue for imitation learning and demonstrate the improvement in a motor cortex-like SNN architecture. The rest of this paper is structured as follows: Sec. 2 introduces the methodology followed for the development of the subnetworks and integration to construct the proposed network; Sec. 3 introduces the results obtained; Sec. 4 gives and discusses the conclusions of this study.

2 METHODS

While the extent of learning through imitation in humans is yet to be fully understood, this study introduces a biologically inspired mechanism to allow the robot improve the quality of the target reaching skill by minimizing deviation from the intended target path. In a previous study \cite{Zahra2020b}, a SNN demonstrated the ability to learn from motor babbling and the ability to build a coarse differential map. While this map allows estimating the motor commands necessary for sensor guided reaching of targets, the coarse estimations leads to wide deviations from the intended path. It was assumed that such deviations arise mainly due to the nature of the training set collected from waving-like motions while moving linearly in joint space. Consequently, providing better training examples, in this case, is one viable solution. In this study, the proposed mechanism acts to not only imitate actions in task space but to learn as well from the activity in joint space to refine the reaching skill. Hence, the joint space of the teaching agent (human arm in this case) is mapped to the joint space of the robotic arm. This mapping correlates the angular positions of the human arm to those of the robotic manipulator which result into the same end effector position (see Fig. 1). Such a correlation in angular positions allows teaching the robot and refining the reaching movements by complementing the training examples by human reaching movements after transforming into the robot’s joint space (i.e., solving the correspondence issue).

2.1 Biologically Inspired Imitation Learning

In this study, a robotic manipulator, with $m$ degrees of freedom (DoF) and a task/action space of $z$ dimensions, executes a target reaching task via low-level joint velocity control. The kinematic relations are built based on data collected from random movements of the manipulator with no prior knowledge of configuration. Hence, the data collected as pairs of sensory readings of the joint space $J S_R (q_r \in \mathbb{R}^m$ and $u_r \in \mathbb{R}^m$) and the task-space $T S (x_r \in \mathbb{R}^z$ and $v_r \in \mathbb{R}^z$) as $m_r^k = \{q_r^{l-1,k}, u_r^{l-1,k}, x_r^{l-1,k}, v_r^{l,k}\}_{t=1,\ldots,T_k}$, where $q_r$ and $u_r$ are the angular position and velocity, respectively, and $x_r$ and $v_r$ are the Cartesian position...
Figure 1. The robotic manipulator and human arm sharing the same end effector position and jointly moving during motor babbling to provide a proper training data.

and velocity, respectively. \( T_k \) is the number of time steps taken to execute the \( k^{th} \) robot reaching movement

\[
\mathcal{M}_r = \{ \{ m_r^k \} \}_{k=1,...,K} \quad \text{where } K \text{ is the total number of movements recorded.}
\]

Such random movements are executed linearly in joint space, which does not normally correspond to linear movements in the task space. Consequently, in most cases, the training data lack for good examples of linear motion in Cartesian-space, which is essential to reduce the time needed for target reaching and to achieve dexterous manipulation. Thus, complementary examples are needed to enrich the training data set. However, it is not possible to generate such examples through robot movements in absence of a mathematical model for the kinematic relations. Hence, it is adequate to provide such examples through a teacher capable of providing the desired movements. It follows that the teacher shall move across the studied \( z \)-dimensional work-space to provide these examples. Although the teacher can have different number of DoFs from that of the robot, in this study, the same number of DoFs is assumed for simplicity. 

So, the human teacher is administered to collect the data from arm joint space \( JS_H (q_h \in \mathbb{R}^m) \) and the task-space \( TS (x_h \in \mathbb{R}^z) \) as \( m_h^k = \{ \{ q_h^{t,k}, x_h^{t,k} \} \}_{t=1,...,T_k} \), where \( q_h \) is the angular position, and \( x_h \) is the Cartesian position. \( T_k \) is the number of time steps taken by the human arm to reach the \( k^{th} \) target. 

\[
\mathcal{M}_h = \{ \{ m_h^k \} \}_{k=1,...,K} \quad \text{where } K \text{ is the total number of targets reached. Then, } \mathcal{M}_h \text{ can be transformed via a separate mapping to the robot coordinates to be utilized in the learning process.}
\]

Thus, to be able to learn the policy \( \mathcal{P} \) mapping the robot configuration to the motor actions, two modes of learning have to be adapted: (i) learning via motor babbling from robot’s own actions \( \mathcal{M}_r \), and (ii) learning by imitating the human teaching agent \( \mathcal{M}_h (P : Q_R \rightarrow U_R) \). The former (i.e., first mode) allows building a generalization of the differential motion achieved for specific motor commands for different configurations \( P : Q_R \rightarrow U_R \) (where \( q_r \in Q_R \) and \( u_r \in U_R \)). While the latter (i.e., second mode) allows refining these motions for specific desired movement paths by transforming \( \mathcal{M}_h \) to the robot’s joint-space \( \Xi : Q_H \rightarrow Q_R \) (where \( q_h \in Q_H \)). The two learning modes are detailed in the following subsection.

### 2.2 Learning via Motor Babbling

To functionally emulate the motor cortex, a spiking neural network is built to transform the intended motion from task-space to motor commands. This motor cortex-like map (MCM) consists of one
Figure 2. Schematic diagrams for MCM

Inert neuron
Highly active neuron
Plastic excitatory connections
Non-plastic inhibitory connections
A2A plastic synaptic connections
Lateral inhibitory connections

Figure 2. Schematic diagrams for MCM

Dimensional arrays of neurons forming input and output layers, with each array encoding either a sensory input value or motor command output as shown in Fig. 2. Input and output layers are connected through all-to-all (A2A) plastic connections obeying the symmetric spike-timing-dependent plasticity (STDP) rule formulated as:

\[ \Delta \epsilon_{ij} = W \left( 1 - \left( \frac{\Delta t}{\tau_a} \right)^2 \right) \exp \left( \frac{|\Delta t|}{\tau_b} \right) \]  

where \( \Delta \epsilon_{ij} \) is the change in the strength of synaptic connection \( \epsilon_{ij} \) connecting the pre-synaptic neuron \( i \) to the post-synaptic neuron \( j \). \( W \) defines the magnitude of the change, the ratio between \( \tau_a \) and \( \tau_b \) defines the window through which change (either increase or decrease) occurs, and \( \Delta t \) is the difference between the timing of spikes at postsynaptic and presynaptic neurons. In the output layer, lateral synaptic connections allow neurons with highest activity to suppress distant neurons for better estimations.

The neurons are modeled as Izhikevich neurons, compromising the computational cost needed and biological plausibility, demonstrated by the ability to reproduce firing patterns of neurons in various brain regions (Izhikevich, 2004). Hence, the adjustment of the parameters in the model allows for better control of the firing dynamics compared to other models. The Izhikevich neuron model is formulated as:

\[ \dot{v} = f(v, u) = 0.04v^2 + 5v + 140 - u + I \]  

\[ \dot{u} = g(v, u) = a(bv - u) \]  

After a spike occurs, the membrane potential is reset as:

\[ \text{if } v \geq 30 \text{ mV, then } v \leftarrow c, \ u \leftarrow (u + d) \]  

where \( v \) is the membrane potential and \( u \) is the membrane recovery variable. The parameter \( a \) determines the time constant for recovery, \( b \) determines the sensitivity to fluctuations below the threshold value, \( c \) gives the value of the membrane potential after a spike is triggered, and \( d \) gives the value of the recovery variable after a spike is triggered. The term \( I \) represents the summation of the external currents introduced.

For the proposed network to execute the desired transformations, the information needs to be input/encoded into the network and extracted/decoded in a proper way. To be able to convert the signals
from and to the network properly, the encoders (converting signals to spikes) and decoders (converting spikes to signals) are used. The input to the sensory layer (during the training and control phases) and motor layers (during the training phase only) is calculated for each neuron based on its preferred (central) value $\psi^c_i$. Thus, the tuning curve for the encoders is chosen to be the Gaussian distribution. The input current to a neuron $i$ for a certain input can be formulated as:

$$\kappa_i = A \exp \left( -\frac{\|\psi - \psi^c_i\|^2}{2\sigma^2} \right)$$  \hspace{1cm} (5)$$

where $\psi$ is the input value, $A$ is the amplitude of the input current, and $\sigma$ is calculated based on the number of neurons per layer $N_l$, and the range of change of the variable to be encoded from $\Psi_{\text{min}}$ to $\Psi_{\text{max}}$. This leads to the contribution of the whole layer to encode a particular value (a process that can be interpreted as “population coding” (Amari et al., 2003)). For input neurons, $\kappa_i$ is the only external current source, while for output neurons, current is injected from both the input layer and the interinhibitory connections in the output layer. The value of $A$ is chosen based on the neuron parameters and different values of activation are assigned for the sensory and motor layers as $A_s$ and $A_m$, respectively. The choice of $A_s$ and $A_m$ along with the neuron parameters allows to have a controlled firing activity and hence a controlled learning process. The developed network acts as a differential map to relate the robot’s current configuration $q_r$ and intended spatial velocity $v$ with the corresponding motor command $u_r$ such that:

$$u_r = g(q_r, v)$$  \hspace{1cm} (6)$$

2.3 A Numerical Simulation: Proof of Concept

To verify the proposed methodology before proceeding to solving the correspondence issue and real robot experiments, a simulation is designed to carry out the verification. A numerical simulation for the reaching task using a 3 link robot is set to compare the results for training using random motor babbling

![Figure 3](image-url)
versus straight path object reaching. First, the forward kinematics for the robot is formulated to describe the relation between joint angles and the end effector position:

\[ x = l_1 c\theta_1 + l_2 c\theta_{12} + l_3 c\theta_{123} \]  
\[ y = l_1 s\theta_1 + l_2 s\theta_{12} + l_3 s\theta_{123} \]  
\[ \Phi = \theta_1 + \theta_2 + \theta_3 \]

where \( \Phi \) describes the orientation of the end effector, \( l_1, l_2, \) and \( l_3 \) define the length of the 3 links starting from the base, \( \Theta = [\theta_1, \theta_2, \theta_3] \) define the joints’ angles as shown in Fig. 3. \( c\theta_i \) and \( s\theta_i \) refer to cosine and sine of \( \theta_i \), respectively, while \( c\theta_{ij} \) refers to cosine of \( \theta_i + \theta_j \), and so on. The Jacobian matrix \( J(\Theta) \) can then be derived to describe the differential relationship between the robot’s joint space and task space:

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{\Phi}
\end{bmatrix} = J(\Theta) \begin{bmatrix}
\dot{\theta}_1 \\
\dot{\theta}_2 \\
\dot{\theta}_3
\end{bmatrix}
\]

By partial differentiation of the differential forward kinematics (DFK) equations, \( J(\Theta) \) can be obtained:

\[
J(\Theta) = \begin{bmatrix}
-l_1 s\theta_1 - l_2 s\theta_{12} - l_3 s\theta_{123} & -l_2 s\theta_{12} - l_3 s\theta_{123} & -l_3 s\theta_{123} \\
l_1 c\theta_1 + l_2 c\theta_{12} + l_3 c\theta_{123} & l_2 c\theta_{12} + l_3 c\theta_{123} & l_3 c\theta_{123}
\end{bmatrix}
\]

To collect motor babbling data, the robot moves linearly in joint space by assigning fixed values for \( \dot{\Theta} \) and substituting to record the corresponding Cartesian position and velocity.

Then, based on equation (10), a differential inverse kinematic solver (DIK) can be built to guide the robot’s motion:

\[
\begin{bmatrix}
\dot{\theta}_1 \\
\dot{\theta}_2 \\
\dot{\theta}_3
\end{bmatrix} = J^\#(\Theta) \begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{\Phi}
\end{bmatrix}
\]

where \( J^\#(\Theta) \) refers to the inverse Jacobian matrix. Through this equation, the simulated robot moves in straight and curved paths by solving for the desired motor commands \( \dot{\Theta} \) to move in a desired direction inside the defined work space. This allows to bypass the correspondence problem and directly verify the efficacy of the main concepts upon which this work is built. Hence, both the collected data sets are used to train the MCM network to demonstrate the improvement achieved in this case, as discussed in the following section.

2.4 Learning by Imitating

To be able to imitate the human teaching agent, it is essential to solve the correspondence issue by transforming the data collected from the agent to the corresponding robot state. Thus, in the studied case, correlation of the joint spaces of both the robot and the teacher at the same position in the task space is carried out. Firstly, a representation of each of the correlated joint spaces is built using a self-organising map (SOM) to allow for dimensionality reduction as shown in Fig. 4. SOM is built upon the rules of competition, cooperation, and adaptation.
Figure 4. Schematic diagrams for SOMs connected through Oja-Hebbian plastic synapses. This architecture allows to correlate the joint spaces of the human arm and robot arm. During training phase, BMUs (in AJ-SOM and RJ-SOM) from both maps that fire together are more likely to have an increase in strength of the connecting synapses. Consequently, during control phase, if the same BMU in AJ-SOM becomes active, the corresponding node in RJ-SOM becomes active as well.

**Competition:** With each node/neuron $k$ associated with a position/weight vector $\omega_k$, the nodes/neurons compete among each other by comparing the weights to that of an introduced data sample $q$. The winning node, known as Best Matching Unit BMU, is chosen to be with the least Euclidean distance between $\omega_k$ and $q$, such that $i = \arg\min_k \|\omega_k - q\|$, where $i$ denotes the index of the BMU. **Adaptation:** The weights vector of the BMU $\omega_i$ is then updated to give a better representation of the input vector $q$. **Cooperation:** While the nodes compete for a given input to be chosen to represent an input vector, the nodes within the neighborhood cooperate to give better estimations of the output. Thus, the nodes within the neighborhood of the BMU are updated as well in the adaptation phase, formulated as:

$$\omega_j(t + 1) = \omega_j(t) + \lambda(t)\eta_{ji}(t)(q - \omega_j(t))$$  \hspace{1cm} (13)

$$\lambda_{ji}(t) = \exp\left(-\frac{||p_j - p_i||^2}{2\varrho^2(t)}\right)$$  \hspace{1cm} (14)

where $p_j$ and $p_i$ are the positions of the $i^{th}$ and $j^{th}$ nodes within the SOM lattice, $\lambda$ is the learning rate, $\eta_{ji}$ is the neighborhood function, and $\varrho$ is the neighborhood radius. Values of the learning rate and neighborhood radius are defined initially at $\varrho_0$ and $\eta_0$, respectively. As the training proceeds for $T_d$, the learning rate and neighborhood radius decay such that:

$$\varrho(t) = \varrho_0 \exp\left(-\frac{t}{T_d}\right) , \eta(t) = \eta_0 \exp\left(-\frac{t}{T_d}\right)$$  \hspace{1cm} (15)

A model of the SOM with varying density of nodes across the map is chosen for this study (Zahra and Navarro-Alarcon [2019]). As the output of the SOM depends on the activity of the neighborhood nodes, this model allows to preserve the quality of the mapping by attracting more nodes closer to the map borders to ensure the presence of enough nodes in the neighborhood for accurate estimations. Thus, the neighbourhood function differs from that of the standard SOM. A coefficient is defined for node density $\varphi$. 

This is a provisional file, not the final typeset article
computed as:

$$\varphi = \exp \left( - \sum_{j \in \Pi} \|w_i - w_j\|^2 \right)$$  \hspace{1cm} (16)$$

where $\Pi$ is the local neighborhood around the node. $\varphi$ allows to quantitatively find the nodes with less number of nodes in the neighborhood, and hence, more nodes shall be attracted to their proximity. Thus, the neighborhood function can be redefined to allow varying the density across the map based on $\varphi$:

$$\eta(t) = \left( \frac{t}{\varphi T_d} \right)^4 \exp \left( -\frac{t}{\varphi(t)^2 T_d} \right)$$  \hspace{1cm} (17)$$

$AJ\text{-}SOM$ and $RJ\text{-}SOM$ provide a representation for human arm joint-space $JS_H$ and robot arm joint-space $JS_R$, respectively. Each SOM is fed with data collected while holding a correspondence between $JS_H$ and $JS_R$, where the training examples are collected while moving in the shared workspace as shown in Fig. 1. The SOMs are trained for several iterations until reaching the target accuracy of encoding for both spaces. Then, the SOMs are connected through Oja-Hebbian synapses and modulated by introducing corresponding samples to both SOMs. The activity $\alpha_j$ of a node $j$ for an input vector $q$ is then decided based on the following equation:

$$\alpha_j(t) = \exp \left( -\frac{\|w_j(t) - q\|^2}{\varphi^2(t)} \right)$$  \hspace{1cm} (18)$$

The synaptic strength is then updated based on the activity of both pre-synaptic $i$ and post-synaptic $j$ neurons:

$$\Omega_{ij}(t + 1) = \Omega_{ij}(t) + \zeta(\alpha_i \alpha_j - \beta_0 \Omega_{ij}(t) \alpha_j^2)$$  \hspace{1cm} (19)$$

$$\beta(t) = \beta_0 \exp \left( \frac{T_d - t}{T_d} \right), \zeta(t) = \zeta_0 \exp \left( \frac{T_d - t}{T_d} \right)$$  \hspace{1cm} (20)$$

where $\Omega_{ij}$ denotes strength of the synaptic connection from node $i$ to node $j$. The terms $\beta$ and $\zeta$ are defined to adjust the learning process by adjusting the $\beta_0$ and $\zeta_0$ coefficients.

This allows for building a static mapping between $JS_H$ and $JS_R$ such that:

$$q_r = f(q_h)$$  \hspace{1cm} (21)$$

where $f$ is the map formed by the described network which allows approximating the value of $q_r$ corresponding to a certain $q_h$ value to give the same end effector position $x$ for both the human and robot agents as shown in Fig. 5. The working space and joint space are chosen to minimize the occurrence of redundant states.

The formed map allows the transformation of the reaching movements demonstrated by the human agent from $JS_H$ to $JS_R$. The angular positions of both agents, the end effector position along with the timestamp are recorded while babbling at a frequency of 100 Hz, which is then downsampled to 30 Hz to allow for significant change between the recorded subsequent points.
Figure 5. A schematic diagram of the correspondence of the human and robot joint spaces along with the task space $TS$. The data collected from the human and robot together allows to build correlation (i.e., $f(q_h)$) between $JS_H$ and $JS_R$. This allows generating more examples to train the map $g(q_r, v)$ correlating $TS$ to $JS_R$ by transforming examples conducted by human arm (in $TS$) into $JS_R$.

2.5 Optimizing the Hyperparameters

In this study, the hyperparameters ($\Gamma$) of MCM are optimized using Bayesian optimization with the the regression model as an adaptive form of tree Parzen estimator (ATPE) (Arsenault, 2018) and the acquisition function as expected improvement (EI). The optimal values for the hyperparameters ($\gamma^*$) are sought through minimizing an objective function $l(\Gamma)$, given by:

$$\gamma^* = \arg \min_{\gamma \in \Gamma} l(\gamma)$$  \hspace{1cm} (22)

A probabilistic regression model gives an approximation of the objective function, defined as $A = P(S|\Gamma)$ to map $\Gamma$ hyperparameters to the likelihood of a score $S$ for the chosen objective function $l$.

The Parzen estimator PE is a kernel-density estimator that relies on a group of continuous distributions/kernels to model some function. TPE is formulated as:

$$P(\Gamma) = \frac{1}{N_k\xi} \sum_{j=1}^{N_k} K \frac{\Gamma - \Gamma_j}{\xi}$$  \hspace{1cm} (23)

where $N_k$ defines the number of the approximation kernels used, $\xi$ is the kernel’s bandwidth, and $K$ is defined as a Gaussian kernel. $U$ and $D$ are modeled to promote hyperparameters with higher likelihood to return lower values for the objective functions for the following observations.

This is a provisional file, not the final typeset article
The EI (Bergstra et al., 2011) can be formulated as:

$$EI_{S^*_i}(\Gamma_i) = \int_{-\infty}^{S^*_i} (S^*_i - S_i) P(S_i | \Gamma_i) dS_i$$  \(24\)

The Bayes rule is applied to replace the posterior \(P(S|\Gamma)\) by \(P(\Gamma|S)\) instead for TPE before substituting in equation \(24\) (Bergstra et al., 2011) to formulate EI as:

$$EI_{S^*_i}(\Gamma_i) = \frac{\mu S^*_i D(\Gamma_i) - D(\Gamma_i) \int_{-\infty}^{S^*_i} P(S_i) dS_i}{\mu D(\Gamma_i) + (1-\mu)U(\Gamma_i)}$$  \(25\)

$$EI_{S^*_i}(\Gamma_i) \propto (\mu + U(\Gamma_i)(1-\mu))^{-1}$$  \(26\)

Hence, it can be concluded that EI maximizes the ratio \(D(\Gamma_i)/U(\Gamma_i)\) to provide better candidates for the search process while maintaining a balance between exploitation and exploration. The reference value of \(S^*_i\) is decided by the value set for the ratio \(P(S_i < S^*_i) = \mu\).

The time complexity for TPE is less than other BO methods (such as Gaussian Process). However, interaction among the hyperparameters is not modeled in TPE. This drawback is addressed in ATPE by concluding from Spearman correlation (Zar, 2005) of the studied hyperparameters the best parameters to tune to explore the search space efficiently. ATPE suggests empirical formulas, taking into account the search spaces’ cardinality, to give optimal values of \(\mu\) and number of candidates needed by the acquisition function to generate a candidate optimal solution (Bergstra et al., 2011).

For the optimization process, the objective function is set to minimize the difference between the target \(v\) and the actual \(\vec{v}\) spatial velocity such that:

$$l(\gamma) = \arccos \left( \frac{\vec{v} \cdot \vec{v}}{||\vec{v}|| ||\vec{v}||} \right)$$  \(27\)

where minimizing the value returned by the objective \(l(\gamma)\) ensures minimizing the error in estimations and hence reduce the deviation from the reference path while reaching a target. The search space for the optimization includes 15 parameters for both the neuronal units and synaptic connections. For the
chosen Izhikevich neuron model, 4 parameters \((a, b, c, \text{ and } d)\) are defined for units in each layer and the
parameters \(A_s\) and \(A_m\) define the the amplitude of the input current to the sensory neurons and motor
neurons, respectively. The other 5 parameters define the synaptic properties for the chosen spike timing
dependent plasticity (STDP) learning rule including the learning rate \(W\), maximum \(C_E\) and minimum \(C_I\)
synaptic weights, \(\tau_a\) and \(\tau_b\).

To train the SNN, examples from both the saved direct motor babbling trails and imitatory transformed
trails are introduced. The motor babbling trails allow the SNN to develop the initial mapping for direct
transformations, while the imitatory trails provide a complementary data set of transformed demonstrations.
Hence, the intended motion paths are demonstrated through the human teaching agent to aid refining the
map formed.

### 3 RESULTS

#### 3.1 Numerical Simulation Results

To test and quantify the improvement achieved by complementing the data sets with direct examples
to reproduce these examples, the simulation, described in subsection 2.3, is employed to test moving in
curved and straight target paths. With the length of the three links set as 30, 30, and 20 cm from base to
deviant and the range of joint angles are set for the base, shoulder, and wrist joints as \([0^\circ, 30^\circ]\), \([20^\circ, 50^\circ]\),
and \([-10^\circ, 30^\circ]\), respectively. To assess the quality of the robot motion, the maximum deviation of the end
effector from the intended path and the ability to reach the target are the chosen metrics. The intended path,
denoted as \(\phi\), is divided into equidistant 1000 points and the actual path, denoted as \(\rho\), is divided similarly
into 1000 points. To check the deviation of each point \(\rho_i\) from the target path, the Euclidean distance to
each point \(\phi_j\) shall be calculated and compared to define the deviation \(\delta_i\) as the least distance measured at
the point \(\rho_i\), such that:
\[
\delta_i = \min_j \|\rho_i - \phi_j\|^2
\]

Thus, the maximum deviation \(\delta_{max}\) for the whole path \(\rho\) is the maximum distance measured for all of its
points, hence:
\[
\delta_{max} = \max_i (\rho_i)
\]

Moreover, the servoing process is considered successful if the arm reaches within a threshold of 1mm away
from the target. The data for moving in a straight line is generated by assigning a target to the robot and,consequently, a vector is concluded from the current position to the target position. The vector, along
with substituting for the current joint angles in \(J^H(\Theta)\) allow calculating the joint velocities necessary to
move in a straight line. The data for moving in curved paths is generated by assigning random joint angles
and moving linearly in the defined joint space. This is equivalent to kinesthetic learning (KL) by guiding
the robot movement manually. The results obtained can be summarized in Table 1 for both \(\phi\) defined as
linear or curved target paths. It can be concluded the feasibility and amount of improvement expected upon
introducing appropriate training data to the MCM network.

#### 3.2 Robot Setup

The human and robot agents are arranged in an adequate setup, as illustrated in Fig. [1] to share the same
end effector position and move jointly in the defined workspace while the robot executes the random motor
babbling. The motion, in this case, is planar utilizing 3 degrees of freedom (DOF) for the agents. By visual
inspection, the human agent stops the motion of the robot when the end effector moves out of the defined workspace or forces a configuration that can not be maintained by the human agent. The human arm is tracked using five aruco markers to be able to extract the angular position of each of the shoulder, elbow, and wrist joints. The posture of the human agent is maintained while collecting the data to fix a reference pose for the base coordinates of the agent.

Two aruco markers are fixed on the arm, two markers fixed on the forearm, and one fixed on the wrist, as shown in Fig. 7. Four vectors are defined to calculate the angular position $q_h$: $\vec{BS}$ extends from the base coordinates and normal to the body, $\vec{SE}$ extends from the first marker to the second one (i.e., along the arm from the shoulder to the elbow), $\vec{EW}$ extends from the third marker to the fourth (i.e., along the forearm from the elbow to the wrist), and $\vec{EN}$ extends from the wrist to the end effector. The angular position then $q_h = [\theta_s^h, \theta_e^h, \theta_w^h]$ can then be calculated as:

$$\theta_s^h = \arccos \left( \frac{\vec{BS} \cdot \vec{SE}}{||\vec{BS}|| ||\vec{SE}||} \right)$$  \hspace{1cm} (30)

<table>
<thead>
<tr>
<th>Maximum deviation</th>
<th>Mean(mm)</th>
<th>Successful trials (out of 10)</th>
</tr>
</thead>
<tbody>
<tr>
<td>linear w/o $KL$</td>
<td>53.7</td>
<td>6</td>
</tr>
<tr>
<td>linear with $KL$</td>
<td>30.1</td>
<td>10</td>
</tr>
<tr>
<td>curved w/o $KL$</td>
<td>40.6</td>
<td>8</td>
</tr>
<tr>
<td>curved with $KL$</td>
<td>15.5</td>
<td>10</td>
</tr>
</tbody>
</table>

Figure 7. (a) The human arm while moving in straight paths and (b) the plot of the Cartesian position of the hand-held object while moving.
The joint encoders provide the angular position of the robotic joints \( q_r = [\theta_{r_1}, \theta_{r_2}, \theta_{r_3}] \). The data collected from human and robot joint spaces are used to train the AJ and RJ SOMs and train the synaptic linkage between them. This linkage allows solving the correspondence issue to provide the complementary examples by transforming the motion executed by the teacher into the robot’s joint space representation to refine the training process in the MCM network. After the training process ends, the end effector of the teacher and the robot is detached to test the performance of the robot in executing the servoing task as demonstrated. The performance metrics are introduced in the next subsections.

### 3.3 Sub-networks Performance

**MCM:** The value of the objective function \( l(\gamma) \) successfully converges to a value of 0.58 rad after around 170 iterations to obtain the values for the network parameters in Table 2. This allows to lower the mean value of the maximum deviation error from around 62 mm, following the tuning method introduced in (Zahra et al., 2020b), to 46 mm in the studied workspace and reduction of the number of neurons per neuron assembly from 136 to 20 neurons. It can be noticed in Fig. 10 that spikes occur in the fitness values, which indicates the balance held between exploration and exploitation while searching for the optimal values.

**SOM:** The mapping of the joint-spaces is studied first using the basic SOM developed by Kohonen, as shown in Fig. 8, to provide a reference value for the improvement in the accuracy of the provided estimations for using the varying density SOM instead, as shown in Fig. 9. The mean error in estimation is concluded to be approximately 0.25 and 0.16 rad in the case of the SOM compared to 0.17 and 0.11 rad in case of varying density SOM for the human and robot agents, respectively. This allows for better estimation of the angular positions and, hence, angular velocities which improves the quality of the training data fed to the MCM.

### 3.4 Target Reaching

With the task to reach targets through a straight line as shortest path, the end effector moves from the current position to a target position, as shown in Fig. 11. Firstly, the data collected from motor babbling is assessed in terms of the mean and standard deviation of the maximum deviation from a straight line. The obtained values for the robot reaching (i.e., reproducing results from (Zahra et al., 2020b)) are 4.2 cm and
2.3 cm for the mean and standard deviation values, respectively, are bigger than those achieved by the human agent while recording the straight line reaching demonstrations with a mean and standard deviation values of 2.1 cm and 1.3 cm, respectively. The teaching imitation data is then generated by introducing these examples to the AJ-SOM and recording the output from RJ-SOM. The mean and standard deviation calculated for these examples to be equal to 3.4 cm and 1.9 cm, respectively, which proves the efficiency of the proposed network and the feasibility of improvement by the generated data.

Different percentages of contribution from the two sets of examples are employed to quantify the enhancement in the reaching movements in each case. Percentages of 30, 50, and 70 are applied with the quality of reaching movements recorded in each case and the results are obtained as shown in table 3.

<table>
<thead>
<tr>
<th>Percentage</th>
<th>Mean (mm)</th>
<th>SD (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>30%</td>
<td>32.7</td>
<td>13.8</td>
</tr>
<tr>
<td>50%</td>
<td>28.1</td>
<td>10.9</td>
</tr>
<tr>
<td>70%</td>
<td>37.3</td>
<td>15.2</td>
</tr>
</tbody>
</table>

Figure 8. Heatmaps of the standard Kohonen SOM depicting the relation between each input from the joint-spaces of (a) the human arm and (b) robot agents.
In this study, the representation capabilities of the SOM and MCM are matched together to allow the robot to reduce the error while reaching targets. The static mapping of spaces by the SOM and the Oja-Hebbian synapses allows transforming human demonstrations into teaching examples in the robot’s joint space. The MCM is trained by examples provided by motor babbling as well as demonstration examples to give the desired results.

Using the varying density SOM reduces the error in static transformation compared to the basic SOM. Additionally, optimizing the parameters, as shown in Table 2 and Fig. 10 of the MCM facilitates decreasing the error in the mapping and reducing the number of neurons in the network compared to relevant previous studies (Zahra et al., 2020b). The proposed method successfully decreases the deviation of the manipulator from the target path: first by applying Bayesian optimization introducing an improvement of around 25%, and the post-optimization deviation is further reduced by 33% through imitation learning. It can be concluded as well that maintaining a good balance of self-generated data and “others” demonstration data helps obtain better results as shown in Table 3. Compared to (Tieck et al., 2017) which utilizes an SNN to imitate grasping actions, the proposed system incorporates a solution for the correspondence issue and attain less error for a wider set of examples.
The proposed system does not take into account handling redundant solutions which shall be considered in future studies. Additionally, the equations ruling the amount and ratio of data from each of these categories shall be further investigated. A spiking model of the SOM shall be employed with a proper optimization technique as well, which would allow utilizing the incorporated temporal domain for faster learning, more biological plausibility, and energy efficient simulation while running in neuromorphic hardware [Evans and Stringer, 2012; Rumbell et al., 2013]. Moreover, combining the cerebellar model with the developed network shall improve the performance and provide good basis for a highly adaptive neural controller (Zahra et al., 2020a; Tolu et al., 2020).

REFERENCES


![Figure 10. Values of the fitness function versus the number of optimization iterations.](image)
Figure 11. (a) The robot arm while moving after training by the transformed data and (b) the plot of the Cartesian position of the end effector while moving.


