A Mirror Neuron Inspired Model for Goal Inference

Floran Stuijt[1]
FloranStuijt@student.ru.nl
Radboud University Nijmegen
[1]Supervised by R.H. Cuijpers & I.G. Sprinkhuizen-Kuyper

December 1, 2008

Abstract

The human mirror neuron system (MNS) refers to brain regions becoming active both when an action is performed, and when a similar action is being observed. Several functions have been attributed to the MNS. It is argued that the MNS is responsible for goal inference, and could be involved in learning by imitation. Here we investigated an existing neural network model which exhibits mirror properties, the Recurrent Neural Network with Parametric Bias (RNNPB). However, this model fails to account for goal inference, when the body of the actor and observer are dissimilar. The RNNPB model is extended such that it is able to explain this general goal inference. This is achieved by ignoring bodily parameters as a direct input to the model, and instead use the perceived state of the world as a body-invariant input for the model.

First, the RNNPB model is tested for robustness against noise and parametric noise in the original signals. It is demonstrated that the model is able to generalize, which means that the extended RNNPB model does not demand strict requirements on the features used for representing the state of the world. Furthermore, we demonstrated the present model in a toy world example, in which it is shown that observation of multiple goal-directed actions and generation of a goal directed action achieving the same goal both leads to equal activation of an additional input layer of the network. This layer contains the so called parametric bias nodes, which is an essential part of the RNNPB architecture. This exemplifies the goal related mirror properties of the model.

1 Introduction

In 1996, Gallese, Fadiga, Fogassi, and Rizzolatti (1996) presented findings of their research performed on macaque monkeys. The aim of this research was to study neurons which are specialized in hand control. However, this research has led to the interesting result that neurons in the F5 region of macaque monkeys discharge both when a goal directed movement is performed, and when a similar movement is being observed.

Neurophysiological (Cochin et al., 1998), and brain imaging (Rizzolatti et al., 1996) experiments resulted in similar findings in the human brain. The inferior frontal cortex and parietal lobe have both been identified as becoming active during execution of a goal directed movement, as well as during observation of a similar movement. Although no single neuron measurements have been performed, it is very likely that these regions contain mirror neurons. These regions are generally referred to as the main areas of the human mirror neuron system (MNS).

Since its discovery, several functions have been attributed to the MNS. Roughly, these functions can be subdivided into three major hypotheses: **i)** *intention understanding*; basically, two theories exist on explaining how humans understand each others' behavior. In the first, 'theory' theory or folk psychology, understanding of other's mental states is explained by attribution of mental states to others by reasoning. The second, contrasting theory, is simulation theory, which states that the observed actor's state is simulated, such that his mental state can be experienced. Since the MNS is active during action observation, while no overt movements can be observed, mirror neurons seem to be involved in simulation of the observed movement. Because of this, mirror neurons have been considered as supporting simulation theory (Gallese & Goldman, 1998). In relation to this, another hypothetical function of the MNS is **ii)** *imitation learning*; learning by imitation is performed by children, as well as by adults. Such a mechanism allows one to map other's behavior to his own motor repertoire, allowing an individual to decrease the time necessary to acquire certain motor skills. Gallese et al. (1996) have postulated this hypothesis as the feature extraction hypothesis. Finally, there is the **iii)** *epiphenomenologists view*, which actually does not attribute any function to the MNS, but assumes that activity of mirror neurons is a by-product of processing in the brain, having no causal effect at all.

Experiments resulted in the finding that mirror neurons do not discharge during random movements, but during goal directed movements, such as grasping for food, or placing of an object (Fogassi et al., 2005). Furthermore, it has been shown that discharge of macaque monkey mirror neurons (Fogassi et al., 2005), and activity in the human MNS (Iacoboni et al., 2005), is associated with intentions of the observed and generated movement, rather than with the actual movements of the involved kinematic bodies. Behavioral experiments (Bekkering, Wohlschlager, & Gattis, 2000) support these findings, since it has been shown that imitation in human beings is guided by goals, rather than actions. These findings support the notion that the MNS is involved in intention understanding (in this article we assume that intention understanding is equal to goal inference. Therefore, in this article these two terms are used interchangeably).

Matching an observed goal-directed movement to execution of motor actions, requires a mechanism responsible for extracting the elements describing the action of the observed agent's action. A technique often used in imitation learning is the mapping of the actor's bodily joints on the observer's kinematic joints. Several MNS models using this approach have been proposed. In this study, we investigate the Recurrent Neural Network with Parametric Bias (RNNPB) (Tani, Ito, & Sugita, 2004; Oztop, Kawato, & Arbib, 2006) as a potential model for goal inference. The RNNPB model is based on a recurrent artificial neural network, extended with a series of input nodes which are referred to as the parametric bias (PB). Since the PB nodes become equally active during generation and recognition of a time series, these nodes exhibit mirror properties.

As noted by Tani et al. (2004), an important issue which is missing in their study of the RNNPB model is the goal-directedness in generating or recognizing behaviors. The imitation learning approach, as used in the RNNPB model, is limited in the sense that it requires similar bodies of actor and observer (Cuijpers, Schie, Koppen, Erlhagen, & Bekkering, 2006). In order to infer the intentions or the goal of the actor, an alternative approach is required. In this article we try to fill this gap by extending the RNNPB model with a goal representation, such that body-similarity problem is circumvented. A model is presented, which is consistent with the experimental findings on the involvement of the MNS in goal inference, while its mirror properties are preserved.

This article is organized as follows. In Section 2, a detailed description of the RNNPB model is given. Next, in Section 3, a description of the presented model is given. Simulations of the RNNPB model and the presented model will be discussed in Section 4. In this section, we will also discuss the inner workings of the models and their strenghts and weaknessess. In Section 5, we finish with a conclusion and a discussion of the presented model.

2 The RNNPB Model

2.1 Overview

The model described in this article is based on a modification of Tani et al.'s Recurrent Neural Network with Parametric Bias (RNNPB) model (Tani et al., 2004). The RNNPB model has a recurrent network architecture, similar to a Jordan type network (Jordan, 1990). Basically, the model consists of a standard three layered feed-forward model, having an input layer, a hidden layer and an output layer. In addition, a context layer provides recurrent connections to the hidden layer. Furthermore, parametric bias (PB) nodes are added which are connected to the hidden layer. A schematic view of this architecture is shown in Figure 1.

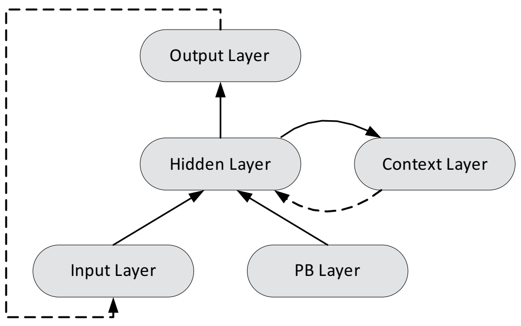


Figure 1: RNNPB model architecture. Arrows between layers denote a full connection of the nodes inside the layer block. Recurrent connections are denoted by dashed lines.

Like a Jordan type network, the RNNPB model is capable of learning to predict a time series. However, during learning, the PB nodes are modulated in a self organizing manner. This provides a way of learning the network to predict multiple time series. If multiple time series are presented during learning, the associated PB node values tend to move to unique values for each provided time series. The PB nodes learn to encode the task the network has to perform.

After the learning phase, the model can be used to generate the learned time series by setting the appropriate PB node values and running the network in closed loop mode (i.e. using the output at time step t as input on time step $t+1$). The major advantage of this model is that after learning it can also be used to recognize the learned time series. Presentation of a time series coerces the PB nodes to take on the same values as during learning of the presented time series. Because of these common activation patterns of the PB nodes during generation and recognition, the RNNPB is said to exhibit mirror properties and the PB nodes are mirror neuron like.

In their article, Tani et al. used the RNNPB network to predict sensory and motor signals. That is, the network learned to predict the next sensory signal $\hat{s}_{t+1}$, and motor command $\hat{m}_{t+1}$, given the current sensory signal s_t and current motor command m_t. In their experiment, a robot was learned to imitate a human agent. This was achieved by mapping the agent's arm position to joint angles of the robot. These angles where extracted from the agent's arm position by putting detectable markers on his arm, and computing joint angles from the position of these markers.

During the learning phase, the robot learned to associate PB node values with the observed joint angle sequences. After learning, an interaction phase was used to demonstrate the model's recognition and generation abilities. During this phase, the robot had to recognize the human agent's movement from the observed joint angle sequences such that the robot was able to perform the observed movement himself. Tani et al. demonstrated this by feeding the observed joint angles to the RNNPB network, and as soon as the PB node values were set, running the network in closed loop generation mode. The resulting state and motor time series were used to drive the effectors of the robot.

For each of the above discussed phases (learning, generation, and recognition), a more elaborate explanation will be given below.

2.1.1 Learning

In the learning phase, the network is trained to predict a given time series $Y = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_{T-1}, \boldsymbol{y}_T\}$. This is accomplished by altering the internal weights of the network (i.e. all weights except for the recurrent connection weights), such that the sum of squared differences E between the output $\hat{\boldsymbol{y}}$ of closed loop execution of the network and the original time series is minimized. Here, output $\hat{\boldsymbol{y}}$ is obtained by forward propagating the node activities through the network. More formally, we want to find the network weights such that

$$E = \sum_{t=1}^{T} ||\hat{\boldsymbol{y}}_t - \boldsymbol{y}_t||^2 \tag{1}$$

is minimized. A multitude of learning algorithms exist to solve this problem, but the one used by Tani et al. is Backpropagation Through Time (BPTT) (Rumelhart & McClelland, 1986). Basically, this learning method is identical to backpropagation, except that the recurrent network is unfolded over time first, such that the network is similar to a feed-forward network*.

For notational convenience, from here on, the collection of node activations within each network layer will be represented by a mathematical vector, and are typeset in bold. Input of the network is denoted by $\boldsymbol{x}$, and output by $\hat{\boldsymbol{y}}$. Activation of the PB nodes is denoted by $\boldsymbol{p}$. The internal values of the PB vector which can be considered as net input to the PB vector, are denoted by $\boldsymbol{u}$.

*Unfolding over time allows one to transform an arbitrary recurrent network into a feed-forward network, extending the applicability of the backpropagation algorithm to networks with an arbitrary amount of recurrent connections.

In order to let the network predict multiple time series, time series are presented to the network in sequential order. Updating of the PB vector takes place after presentation of each time series. After all the time series have been presented, the network's internal weights are updated according to the BPTT algorithm. This cycle, from here on referred to as an epoch, is repeated for a predefined amount of times.

In each learning epoch, the network is run in closed loop mode. The recurrent input $\boldsymbol{x}_{t+1}$ at time $t+1$, used in the closed loop generation, is calculated as follows:

$$\boldsymbol{x}_{t+1} = \beta \hat{\boldsymbol{y}}_t + (1-\beta)\boldsymbol{y}_t, \tag{2}$$

where $\boldsymbol{y}_t$ and $\hat{\boldsymbol{y}}_t$ respectively represent the target signal and the output activity at time step t. β represents the input ratio, which is used to combine the expected signal and actual signal in order to gain acceptable learning performance[†].

The following update rules are used to modify the PB vector during learning[‡]:

$$\Delta \boldsymbol{u}_e = \eta \sum_{t=1}^{T} \boldsymbol{\delta}_t, \tag{3}$$

where $\boldsymbol{\delta}_t$ represents the back propagated error for the PB vector at time step t, $\boldsymbol{u}_e$ represents the internal values of the PB vector at epoch e, and η is the PB learning rate. The PB vector values are updated as follows:

$$\boldsymbol{u}_{e+1} = \boldsymbol{u}_e + \Delta \boldsymbol{u}_e. \tag{4}$$

Finally, the PB vector is related to the internal values by

$$\boldsymbol{p}_{e+1} = g(\boldsymbol{u}_{e+1}), \tag{5}$$

where $g(\cdot)$ is the standard logistic activation function, given by

$$g(x) = \frac{1}{(1+e^{-x})}. \tag{6}$$

2.1.2 Generation

In the generation phase, the network is used to generate the time series associated with the provided PB vector during learning. This is accomplished by setting the PB vector to the value associated with the desired time series, and executing the network in closed loop for a desired amount of time steps. During this phase, no updating of internal weights and PB vector values take place. The output is generated solely by employing forward propagation of the node activities through the network.

2.1.3 Recognition

The recognition phase is similar to the learning phase with the exception that the internal weights of the network are not updated. Consequently, only the PB vector is updated during this phase. As a result the PB vector takes on the same value as during the learning phase.

[†]This method is also known as teacher forcing (Jaeger, 2002).

[‡]In their original article, Tani et al. (2004) use ρ instead of $\boldsymbol{u}$ to refer to the interval values of the PB vector. Moreover, p was used to refer to the PB vector. For clarity, an alternative notation is used in this article.

3 Goal Inference using the RNNPB Model

In Tani et al.'s experiment, observed motor angles were mapped onto the actor's body, such that generated motor/state sequences could be used during observation and generation of bodily movements. Therefore, the RNNPB model is unable to explain goal inference due to the fact that observed motor actions directly serve as input to the network. As a consequence, the network is sensitive to the actual movements. Clearly recognition is best when the body of the observer and actor are similar.

In this article a novel approach is proposed, in which both actions and states are represented as separate sequences. This allows observation of goal related movements without referring to a body which manipulates the specific object in a goal oriented fashion. If the agent has learned to associate motor actions to a series of states (forward model), the goal can be achieved by executing these motor actions. On the contrary, by using an inverse model, actions can be associated to a desired state. A schematic overview of this approach, in relation to the approach of Tani and colleagues (Tani et al., 2004), is given in Figure 2(a) and Figure 2(b).

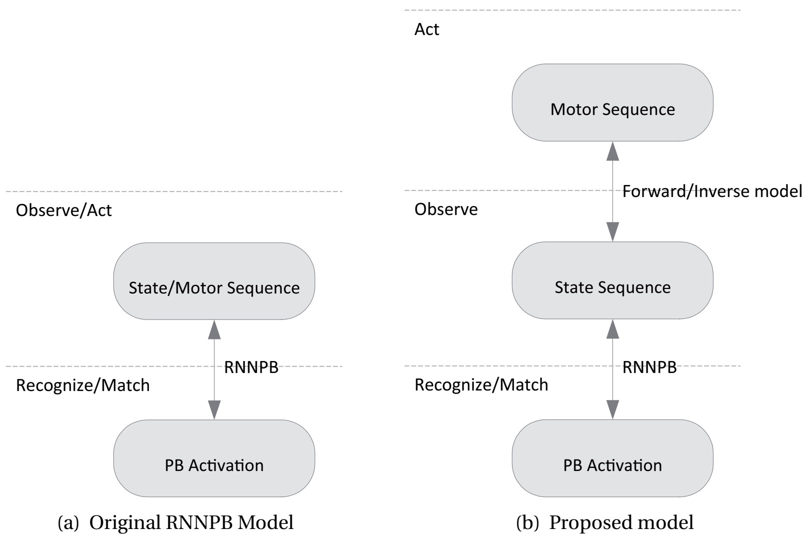


Figure 2: A comparison of Tani et al.'s model and the proposed model. An additional layer has been added to the original RNNPB model, removing the tight coupling between observed motor sequences and PB node activation. This gives the model the ability to be used for goal inference.

Separation of action and state has two major benefits in comparison to the original RNNPB model. First, it does not require similarity between the body of the actor and observer, because states are, by definition, invariant for differences in kinematics. Consequently, it allows goal inference even when the observed motor sequences differ from the observers own way of achieving the same goal. This gives the model the ability to be used in intention understanding and learning by imitation.

The model implements the above as follows. As soon as the agent senses that a goal has been achieved, he is triggered to learn the past state sequence. By utilizing the RNNPB network, the resulting PB vector is associated with the achieved goal described by the presented state sequence. Note that this requires a form of working memory, responsible for retaining the past states.

In order to recognize a goal oriented action, the perceived state sequence is fed directly to the network. As a consequence, the PB vector should converge to the value associated with the presented state sequence during learning. Using the resulting PB vector, the perceived state sequence can be associated with the goal or intention of the actor who caused the perceived changes in state.

4 Simulation and Results

4.1 Mirror Properties of RNNPB Networks

In order to simulate the modified model, we define three unique signals, and repeat Tani et al.'s simulations using these signals. The following signals are used:

1. Sinusoidal time series ($\frac{1}{2}(0.8\sin(3t)+1)$),
2. Same as 1. but with doubled frequency ($\frac{1}{2}(0.8\sin(6t)+1)$), and
3. Aperiodic time series ($0.9-0.8e^{-t}$)).

The following network architecture is used: 1 input node, 1 output node, 5 hidden nodes, 2 context nodes, and 2 PB nodes. The following learning parameters are used: $\eta = 0.01, \beta = 0.1, \eta_{BP} = 0.02$, and $\alpha = 0.9$, where η_{BP} and α respectively denote the learning rate and momentum parameter for the BPTT algorithm. These parameters were determined experimentally.

Results of simultaneously training a single network with these signals are shown in Figure 3. In this figure, the original time series are shown in the left column, and the resulting patterns of running the network in closed loop mode (generation phase) are shown in the right column. The PB vectors are denoted by $\boldsymbol{p}_k$, where k corresponds to the signal number (e.g. $\boldsymbol{p}_3$ denotes the PB vector associated with the aperiodic signal). As can be seen in the figure is that the network successfully reproduces all three time series.

Furthermore, the network recognizes all three signals successfully (see Figure 4). In the left column, the modulation of the PB vector during learning is shown. Note that the latest known value of the PB vector, obtained at the end of the learning phase, is associated with the corresponding time series. The right column shows the modulation during the recognition phase. As expected, we can see that the PB vectors converge to the values that where associated with the presented time series during learning. However, in case of the third signal, the PB vector did not seem to stabilize. This was not a problem, because training and recognition was performed for an emperical determined amount of 2600 epochs. Selection of a stopping criterion for learning and recognition is outside the scope of this article, but for an overview and discussion on various stopping criteria, see Haykin (1998).

During the learning phase, it was observed that learning using BPTT is a very delicate process; if learning is performed too long, the learning process becomes unstable and the weight alterations become too high, pushing the network into a state from which it is unable to recover.

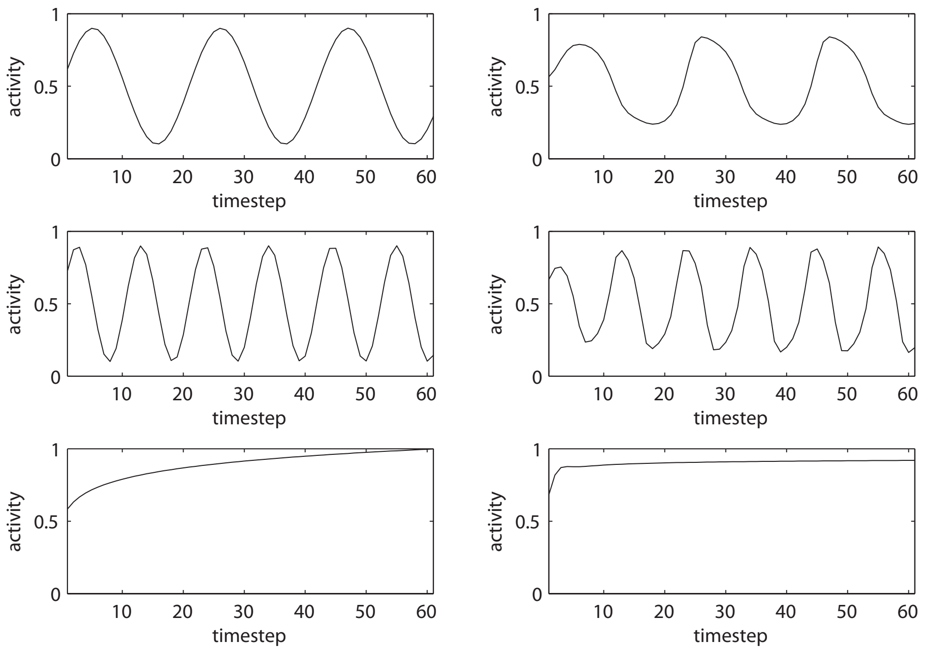


Figure 3: The network's ability to generate a time series in closed loop mode (right column) after simultaneously learning the corresponding signals (left column). Each row number corresponds to one of the signals as described in Section 4.1.

4.2 Generalization and Robustness

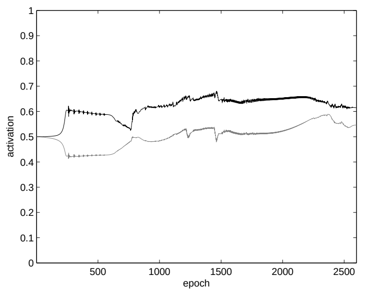
Additional simulations were run to test the model for robustness against noise and parametric noise in the presented signals during recognition. This allowed us to analyze the models' generalization abilities.

First, the original signals were distorted by adding Gaussian noise to them. Secondly, the model we tested is capable to recognize signals of varying amplitude and frequency. For each measurement, the PB vector resulting from recognition were compared to the values that were learned. The error ε of the resulting PB vector was measured using the Euclidean distance metric,

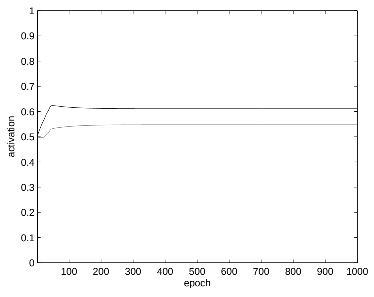
$$\varepsilon = ||\boldsymbol{p}_{target} - \boldsymbol{p}_{actual}||. \tag{7}$$

This error was then calculated for various noise levels. Figure 5 shows the results of this simulation. In this figure, the error of the resulting PB vector is plotted as a function of the amount of noise (i.e. standard deviation σ of Gaussian noise) in the presented signal. This is shown for each time series. The solid line in the figure represents the error curve of the expected PB vector and the resulting PB vector. In order to demonstrate that the network is still able to discriminate the presented time series from the other learned time series, the difference between the other learned PB vectors and the resulting PB vector is also plotted. It can be seen that after adding a fair amount of noise ($\sigma < 0.18$) to the presented signal, the network is still able to successfully discriminate between the learned time series.

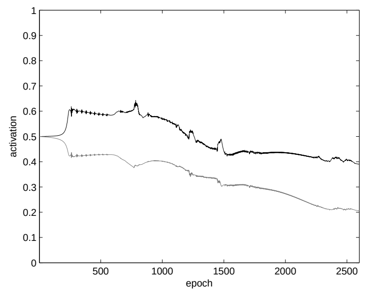
The network is most robust with respect to noise in the high frequency signal. A possible explanation is that noise has a high frequency, and, thus forces the network



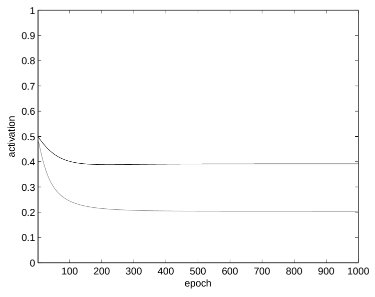
(a) $\boldsymbol{p}_1$ training



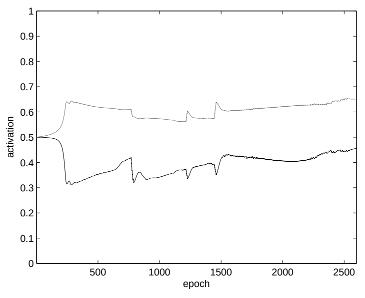
(b) $\boldsymbol{p}_1$ recognition



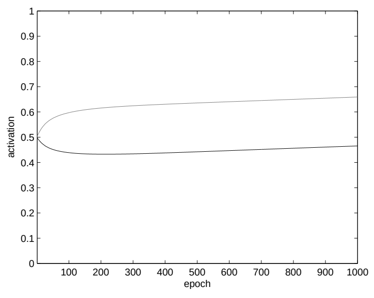
(c) $\boldsymbol{p}_2$ training



(d) $\boldsymbol{p}_2$ recognition

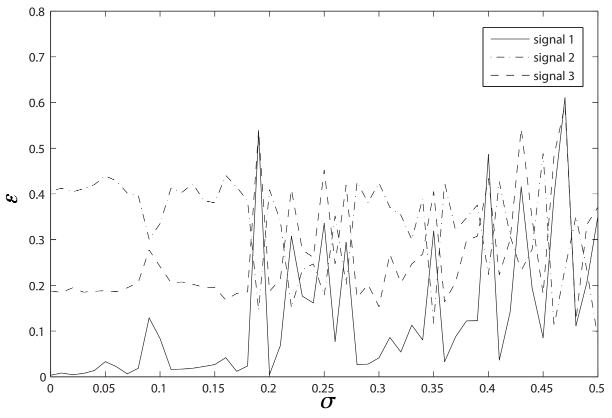


(e) $\boldsymbol{p}_3$ training

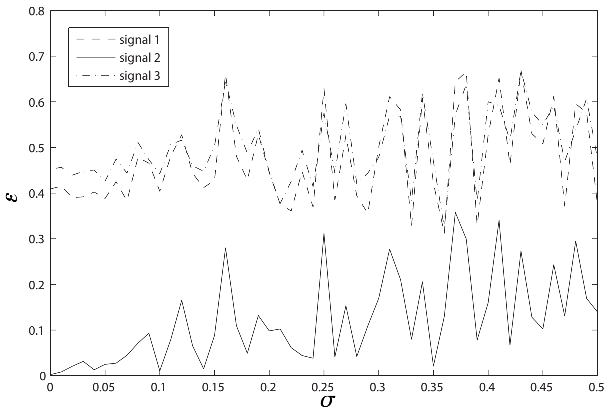


(f) $\boldsymbol{p}_3$ recognition

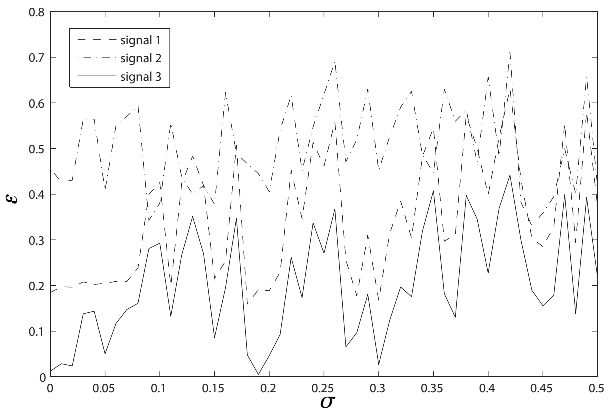
Figure 4: PB vector modulation during training (left column), and during recognition of the corresponding signal (right column). It can be seen that during recognition, the PB vectors converge to the values associated with the presented time series during learning.



(a) Distortion of signal #1



(b) Distortion of signal #2



(c) Distortion of signal #3

Figure 5: Error between the recognized PB vector and each known PB vector after presenting the network a Gaussian distorted version of the original time. It can be seen that the associated PB vector is still the best matching vector in terms of Euclidean distance ε, after adding a fair amount of noise (e.g. $\sigma < 0.18$ for the first signal).

to the PB vector associated with the high frequency signal. However, during recognition of the other noisy signals, no attraction to the high frequency PB vector can be observed.

Figure 6 shows the error ε between the recognized PB values and the trained PB values, as a function of amplitude and frequency of the presented signal. Due to the third signal's aperiodic nature, the parametric noise simulation was only applied to the first and second signal.

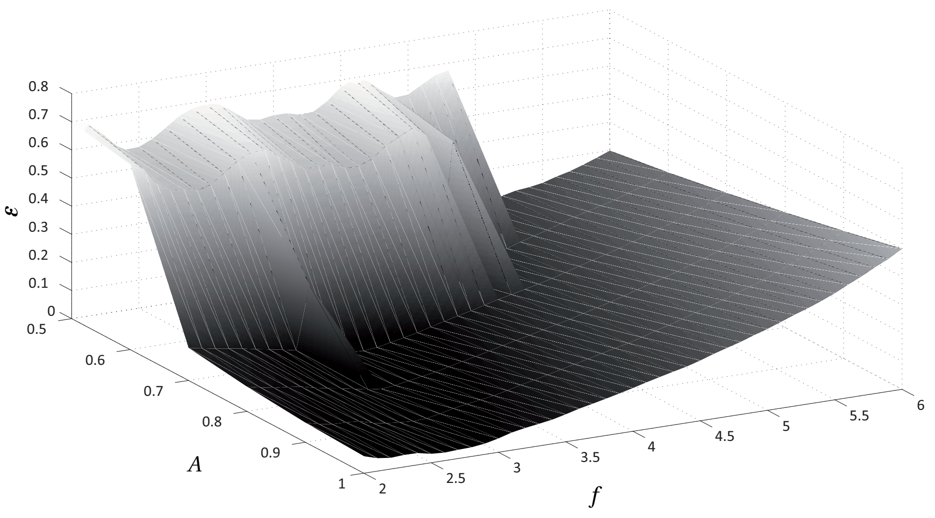


Figure 6: This figure illustrates the model's sensitivity to alterations of the original signal, by plotting the Euclidean error between the expected PB vector and resulting PB vector as a function of amplitude A and frequency f. Here, the expected PB vector is equal to the PB vector associated with the single frequency sinusoidal signal. Comparison of the other PB vectors yielded similar results.

As can be seen in the figure, the global minimum lies as expected on the original amplitude and frequency values, while the error slightly increases when moving away from this spot. Similar results were obtained for the second signal (not shown).

Another way to look at the network's performance is by plotting the indidual components of the PB vector, for varying frequency and constant amplitude, and vice versa (see Figure 7). For the presented signal, a frequency interval of 2Hz to 7Hz was used, while an amplitude interval of 0.5 to 1 was used. For both intervals, a stepsize of respectively 0.1 and 0.05 was applied. If the network has learned to encode a property of the trained time series, we would expect that the components of the resulting PB vector follow a smooth trajectory in the plane $\mathbb{R}^2$ when a parameter of the presented time series is increased or decreased. As can be seen in the figure, in case of the frequency-variation, the PB vector follows a smooth trajectory in the plane (the trajectory denoted by the gray dots). This indicates that the network has learned to implicitly represent the frequency component of the presented signal in the PB vector. On the other hand, generalization of the amplitude is limited, as can be seen by the discontinuity of the trajectory denoted by the triangles. The encircled values in the figure correspond to the PB vector values associated with the learned time series.

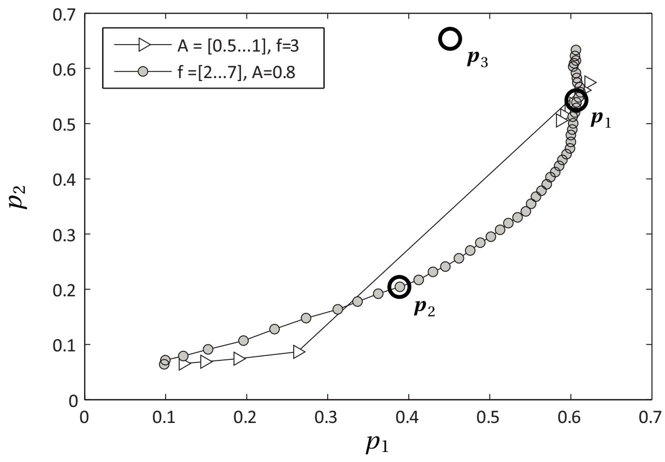


Figure 7: A 2D geometrical interpretation of the PB vector $\boldsymbol{p} = [p_1, p_2]^T$, obtained as a result of varying the frequency f of the presented signal, while keeping the amplitude A constant (denoted by the gray dot marker). The contrary case is also shown, in which frequency was kept constant, while amplitude is varied (denoted by the triangle marker). The latter case was performed only for the single frequency sinusoid. A frequency interval of 2Hz to 7Hz was used, while an amplitude interval of 0.5 to 1 was used, using a stepsize of respectively 0.1 and 0.05. The encircled PB vectors $\boldsymbol{p}_k$ correspond the learned PB vectors associated with time series k.

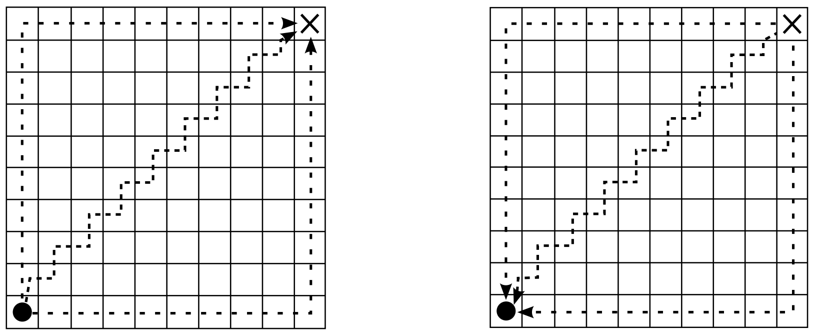
4.3 Model Demonstration

After showing that the model is able to both generate and recognize time series, we can illustrate the main purpose of our model. To do so, A toy world has been constructed, with the following features:

- a set of reachable goals,
- a set of perceivable features (states), and
- a set of possible actions.

In the world used in this experiment, two goals have been defined (see Figure 8): **i)** moving from the location with the dot to the location marked by the cross, and **ii)** moving from the location marked by the cross to the location marked by the dot. The set of perceivable features consists solely of the ‘distance to the cross’ feature. This feature is calculated as the Euclidean distance between the current board position and the cross. The set of possible actions consists of moving up, down, left, and right.

For each goal, three ways of achieving it have been simulated, and the resulting state sequence was stored. Next, the model was trained to learn to predict one of the three time series for both goals. This is illustrated in Figure 9. The left column shows the time series used to train the network. Here, one of the non-diagonal movements was used. Both non-diagonal movements produce the same sequence. Thus, the network is already trained for two different ways of achieving the goal. The right column shows the network’s ability to reproduce the trained signal.



(a) Goal #1 (b) Goal #2

Figure 8: The board used in simulation. Two goals can be distinguished, namely a) reaching the cross from the dot, and b) reaching the dot from the cross. The dashed arrows denote alternative paths which can be followed to achieve the goals. These paths are constructed using the available motor actions, which are moving up, down, left, and right.

After learning, recognition of different ways of achieving the defined goals was tested. During recognition, the diagonal alternative of both goal directed state sequences were used as input. For both sequences, recognition yielded correct PB vector values ($\varepsilon < 0.01$).

By this simulation it is shown that goals of an observed action can be inferred, even if the bodies are dissimilar, and if an alternative way of achieving the same goal is used. This demonstrates the model's invariance for differences in kinematic bodies, and different ways in which a goal can be achieved.

5 Conclusion and Discussion

Ever since it's discovery, several functions have been attributed to the Mirror Neuron System (MNS). Several models for the MNS have been proposed. The RNNPB model, as proposed by Tani et al. (2004), is used as a basis for the present model. However, the original RNNPB model lacks the ability of general goal inference, since it requires that both the body of the observer and actor are similar. Furthermore, it does not account for different ways in which a single goal may be achieved.

A model has been constructed, based on Tani et al.'s RNNPB model. Unlike Tani et al.'s original model, this model is capable of goal inference, while its mirror properties have been preserved. The present model circumvents the body-similarity problem by separating recognition of motor and state sequences, which are tightly coupled in the RNNPB model. The presented model provides a theoretical foundation for experimental findings, which state that the MNS is involved in goal inference. Simulations have been performed to illustrate this functionality.

Simulation have shown that the model is robust against parametric variations in the presented state sequences. This property allows one to select features which are not fully invariant over different means in which a goal may be achieved. Furthermore, it has been demonstrated that the model is robust against noise, which is a must-have for use real world applications.

Further simulations demonstrated the model's ability to recognize a goal, while

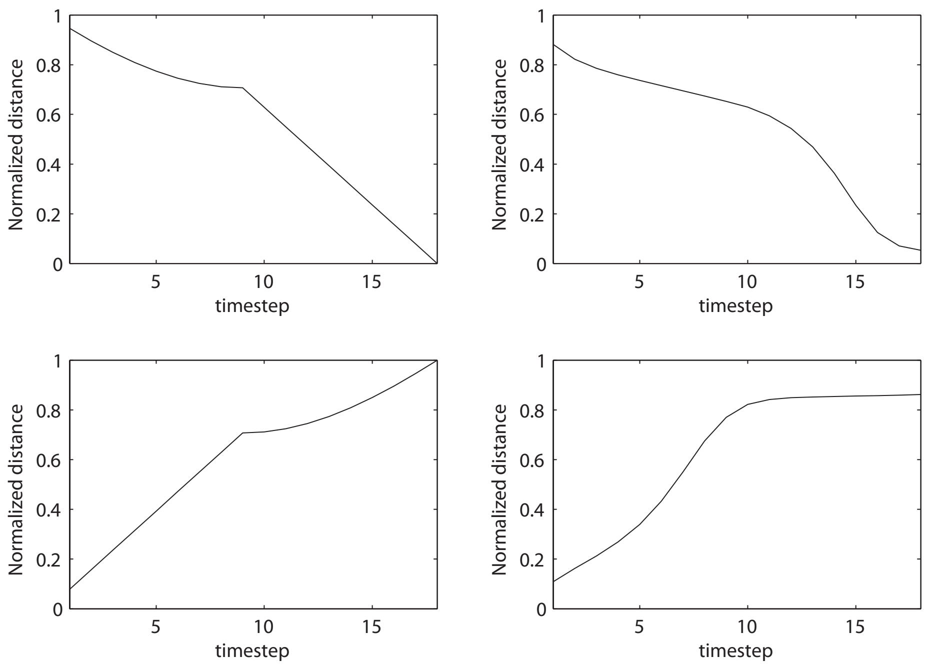


Figure 9: Results of training the model on the selected time series representing two goal-directed movements, each of them directed to a different goal. The first row shows the time series associated with the 'move from dot to cross' goal. The second row shows the time series associated with the 'move from cross to dot' goal. The left column shows the time series used as input during learning, and the right column shows the result of running the network in closed loop mode after the learning phase.

the means of achieving the goal are varied. This makes the model suitable for recognizing and adapting to new ways in which a goal could be achieved, allowing the model to optimize its ways of achieving goals. This is consistent with experimental findings, which support the notion that the MNS is involved in intention understanding and imitation learning.

The generation of motor commands from a generated sequence of feature values was left out the scope of this article. People that are interested, can implement such a mapping by creating a forward and inverse model using a look-up table. Such a look-up would simply map from desired states to necessary motor actions, and vice versa.

Some questions are left open for further investigation. First of all, no research has been done on stopping criteria for learning and recognition. Some proposals have been made in (Haykin, 1998). Furthermore, no research has been performed on the learning algorithm, leaving questions open regarding, for example, stability of learning.

References

Bekkering, H., Wohlschlager, A., & Gattis, M. (2000). Imitation of Gestures in Children is Goal-directed. *The Quarterly Journal of Experimental Psychology Section A*, *53*(1), 153–164.

Cochin, S., Barthelemy, C., Lejeune, B., Roux, S., & Martineau, J. (1998). Perception of motion and qEEG activity in human adults. *Electroencephalography and Clinical Neurophysiology*, *107*(4), 287–295.

Cuijpers, R., Schie, H., Koppen, M., Erlhagen, W., & Bekkering, H. (2006). Goals and means in action observation: A computational approach. *Neural Networks*, *19*(3), 311–322.

Fogassi, L., Ferrari, P., Gesierich, B., Rozzi, S., Chersi, F., & Rizzolatti, G. (2005). Parietal Lobe: From Action Organization to Intention Understanding. *Science*, *308*(5722), 662–667.

Gallese, V., Fadiga, L., Fogassi, L., & Rizzolatti, G. (1996). Action recognition in the premotor cortex. *Brain*, *119*(2), 593–609.

Gallese, V., & Goldman, A. (1998). Mirror neurons and the simulation theory of mind-reading. *Trends in Cognitive Sciences*, *2*(12), 493–501.

Haykin, S. (1998). *Neural Networks: A Comprehensive Foundation.* Prentice Hall PTR Upper Saddle River, NJ, USA.

Iacoboni, M., Molnar-Szakacs, I., Gallese, V., Buccino, G., Mazziotta, J., & Rizzolatti, G. (2005). Grasping the intentions of others with oneŠs own mirror neuron system. *PLoS Biol*, *3*(3), e79.

Jaeger, H. (2002). *Tutorial on Training Recurrent Neural Networks, Covering BPPT, RTRL, EKF and the" echo State Network" Approach.* GMD-Forschungszentrum Informationstechnik.

Jordan, M. (1990). Attractor dynamics and parallelism in a connectionist sequential machine. *IEEE Computer Society Neural Networks Technology Series*, 112–127.

Oztop, E., Kawato, M., & Arbib, M. (2006). Mirror neurons and imitation: A computationally guided review. *Neural Networks*, *19*(3), 254–271.

Rizzolatti, G., Fadiga, L., Gallese, V., & Fogassi, L. (1996). Premotor cortex and the recognition of motor actions. *Cognitive Brain Research*, *3*(2), 131–141.

Rumelhart, D., & McClelland, J. (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition.* MIT Press.

Tani, J., Ito, M., & Sugita, Y. (2004). Self-organization of distributedly represented multiple behavior schemata in a mirror system: reviews of robot experiments using RNNPB. *Neural Networks*, *17*(8-9), 1273–1289.