Real-time Gesture Mapping in Pd Environment using Neural Networks

Arshia Cont
Researcher, La kitchen
78 Avenue de la République
75011, Paris, France.
Tel: +(33) 1 56 79 02 89
Arshia.Cont@la-kitchen.fr

Thierry Coduys
Director, La kitchen
78 Avenue de la République
75011, Paris, France.
Tel: +(33) 1 56 79 02 89
Thierry.Coduys@la-kitchen.fr

Cyrille Henry
Researcher, La kitchen
78 Avenue de la République
75011, Paris, France.
Tel: +(33) 1 56 79 02 89
Cyrille.Henry@la-kitchen.fr

ABSTRACT

In this paper, we describe an adaptive approach to gesture mapping for musical applications which serves as a mapping system for music instrument design. A neural network approach is chosen for this goal and all the required interfaces and abstractions are developed and demonstrated in the Pure Data environment. In this paper, we will focus on neural network representation and implementation in a real-time musical environment. This adaptive mapping is evaluated in different static and dynamic situations by a network of sensors sampled at a rate of 200Hz in real-time. Finally, some remarks are given on the network design and future works.

Keywords

Real-time gesture control, adaptive interfaces, Sensor and actuator technologies for musical applications, Musical mapping algorithms and intelligent controllers, Pure Data.

1. INTRODUCTION

Gestural control of musical events has become a trend in computer music over the past years. Rapid growth of sensor technologies and their processing tools in addition to the growing computing power of personal computers have made this technology available to the computer and electronic music communities. On the other hand, most sensor mapping approaches are fixed and confined to few parameters thereby do not allow much control and freedom over musical events. On the other hand, from an instrumental view, the problem of mapping is essential for any new instrument design and the rapid growth of technology is asking for new and intelligent mapping algorithms and controllers.

Our aim was to implement an intelligent black-box that adapts to gestures performed by the user. The black-box would accept parameters describing physical gesture as input and generates the desired high-level parameters as output. For this augmented mechanism of mapping, we have chosen adaptive neural networks which would learn automatically during a training phase.

Comparing to conventional methods of mapping, this system has the following advantages:

- The user can work directly with the desirability of correspondence between gesture and produced results, rather than the complex mechanism of the mapping algorithm
- The empirical approach of neural networks can evade the complexity of formalizing the problem
- The system can perform well even in the presence of non-linearity and noise in the input
- Ability to make mappings of 'unseen' input patterns
- Cheap computation compared to other methods of complex mappings
- Neural Networks do not require expertise to train and maintain the network.

2. HISTORICAL REMARKS

Among works in adaptive interfaces for musical control, we cite two that have most influenced our approach.

A Neural Network Interface between a Data-Glove and a Speech Synthesizer has been implemented by Sidney S. Fels and G. Hinton in University of Toronto [1]. This system recognizes the hand gestures of sign-language using adaptive neural networks. The networks' output is the parameters for a speech synthesizer. The trained networks are small enough to achieve real-time performance, however the training sessions are heavy and the system was not developed in a musical environment.

In another approach and concerned with the other aspect of adaptive interfaces, i.e. the training, the Center for New Music and Audio Technologies has implemented a simple back propagation with forward-pass as an external (called "mlp", short for "multi-layer perceptron") for Max/MSP programming environment [2]. This implementation is among the first of such interfaces in a real-time musical environment and the graphical interface allows the user to train and obtain a network with ease.

The main interest of Glove-Talk for our application is the Neural Network concept and architecture for the gesture recognition. The interest of CNMAT's "mlp" object lies in the training interface implemented in Max/MSP real-time programming environment.

3. OVERVIEW OF THE SYSTEM

The system consists of two different stages: the first being the network design and the second the real-time application.

In the first stage—training and design of the network—we use Neural Network simulators such as SNNS and Matlab's Neural Network Toolbox. Software developed at La kitchen takes the networks designed in these simulators and produces text files that consequently serves as input for the Neural Network abstractions in Pure Data real-time music programming environment [5]. Our final goal is to implement a neural network training environment in Pure Data itself that handles several architectures and network schemes useful for instrumental and gestural system design.

During this research, several network architectures and training schemes were evaluated and those suitable for general musical applications were implemented for further use. We experimented with two general architectures: Static-networks and Dynamic-networks. In a static network, we map the data without any use of time-delay for recognition and in dynamic application we use time-delay that introduces a notion of memory in a pattern recognition problem and allows gesture mapping and control.

In the second stage, data acquisition is done using series of sensors that describe gestures. A sensor interface routes this data to a real-time software environment containing a trained neural network. Finally the neural network produces the desired parameters for event generation and continuous control of a musical process or application. A well-trained neural network system in general, would generalize its knowledge and respond to unseen gestures appropriate to defined preferences.

The interest of the adaptable mapping lies in the ability to use any input device for mapping. For the sake of this research, data acquisition is done using La kitchen's sensors and the "Toaster" or the wireless "Kroonde" interfaces [3] (see Figure1). These interfaces provide the gesture data to a real-time musical application using the OpenSoundControl protocol [4]. Preprocessing operations designed with the interface and sensors, prepare the desired inputs for the trained network. Sensors and interfaces designed in La kitchen allow a time resolution input of 5ms (equivalent of 200Hz) with 10-bit precision for "Kroonde" and 16-bit precision for "Toaster" which would allow high-resolution control over the events.

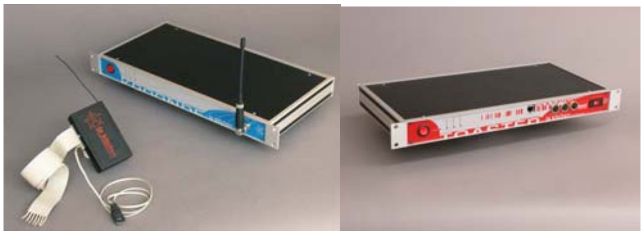

Figure 1. Kroonde and Toaster, high-resolution real-time sensor interfaces.

For the real-time musical environment, Pure Data [5] is chosen for its availability, open-source license and its active community. Data acquisition patches were already available for the mentioned sensors and interfaces along with some preprocessing tools in Pure Data developed at La kitchen. Neural network abstractions are created in Pure Data which take a trained network and perform the network operations. For their design, we had the following constraints in mind:

- High-level and easy interface
- General network implementation that minimizes the sophistication of huge networks
- Real-time performance

4. NEURAL NETWORK REALIZATION IN PURE DATA

A general neuron is presented in Figure 2. A designed network consists of a series of additions and multiplications along with a transfer function. Each neuron's operation can be considered as vector operations.

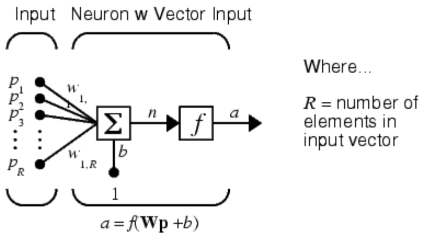

Figure 2. A neuron model in a neural network

A neural network is made up of layers, each of which contains several neurons. With the above model, problems arise when constructing more complicated networks that demand a large number of neurons. To solve this sophistication, we consider each layer as a matrix operation as shown in Figure 3. In this approach, the number of neurons used depends on the size of the input vector and weight matrix and all operations including the transfer function would be on matrices instead of numbers. Most matrix operations for this purpose are available through Pure Data's Zexy library.

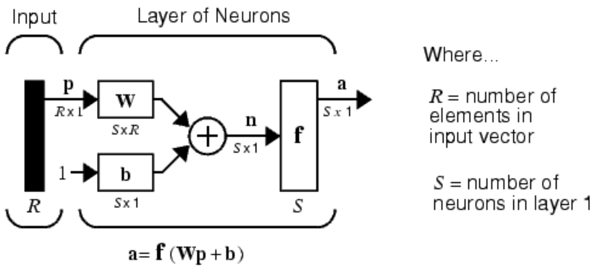

Figure 3. Matrix representation of a layer in a neural network

Figure 4 shows a Pd realization of a layer in neural network. Each layer, despite the number of neurons will be represented by one abstraction and reduces the complexity of network representation. The left inlet accepts the input arranged as a matrix and other inlets would be the trained network parameters which are: layer's weight matrix (equivalent of **w** in Figure 5), layer's bias vector (equivalent of **b** in Figure 5) and the transfer function. These trained parameters are loaded only once into the patch. The transfer function is selected by sending a symbol to the inlet which can be either a log-sigmoid or tan-sigmoid functions. Due high-level representation of network parameters, more transfer functions can be defined by the interested user. In our applications so far, we have not encountered any need for others. The two mentioned transfer functions output data in a range of [0,1] and [-1,+1] respectively which is useful to know for designing the network.

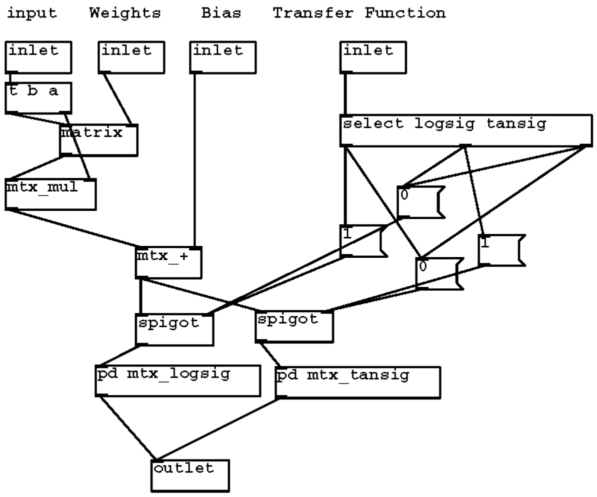

Figure 4. Pd abstraction of a layer of a neural network

For time-delay considerations- to be elaborated in the following sections- we prepare the inputs of a neural network using down sampling and constructing delay lines.

Using the above abstraction, we have constructed a time-delay feedforward neural network which can be used for mapping and gestural recognition systems. There is only one mathematical function which is written as an external for PD. The high-level aspect of this realization allows easy modification of the architecture and implementation of other architectures using the same concept. Figure 5 shows the details of a two layer neural network which will be used as a single abstraction in a mapping system patch.

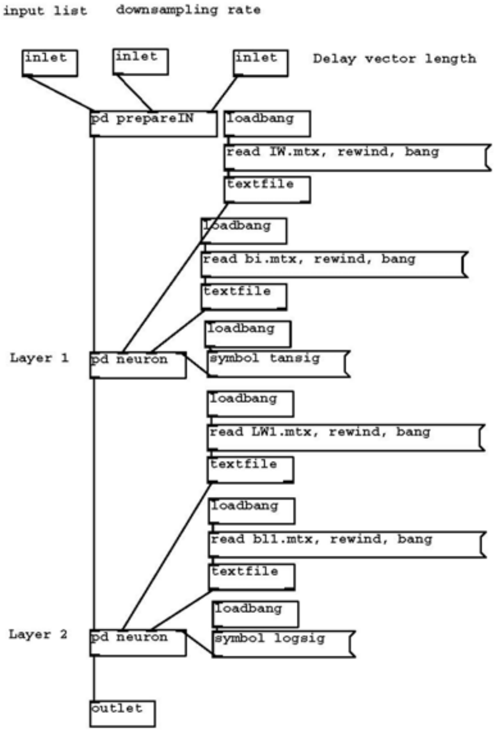

Figure 5. Details of a two-layer network patch

5. TRAINING THE NEURAL NETWORK

Network design and training is the first step towards constructing a network for real-time mapping. For this, the user will need some predetermined input samples, their targets and a defined architecture to start the training.

The training session of a neural network does not require an expertise to realize the network. An empiric approach with trials and errors would eventually make the network converge to the desired behavior. However, a clever choice of network architecture and parameters would save a lot of time in realizing the network.

In our experience, for most musical applications, a maximum of three hidden layer seems to suffice. While use of excessive neurons makes the network converge more rapidly, it degrades the generalization in most applications. There is no law determining the number of neurons needed; this factor is realized by a heuristic approach towards the design of neural networks.

There are numerous learning methods available in the neural network literature. For sensor mapping applications, batch training suffices for convergence. However due to high non-linearity of sensors, the most convenient method does not necessarily converge. We have found that for most sensor mapping applications, the Reduced-Memory-Levenberque-Marquardt algorithm [6] converges with acceptable generalization.

After the training is done and results are satisfactory, network weights and parameters are automatically prepared as Pd MTX files to be used in the real-time application.

6. REAL-TIME APPLICATION

At this point, we will examine the neural network mapping for two different applications: a static network which would map a sensor network to a simple spherical coordinate and a dynamic network for pattern recognition of a circular movement using a network of sensors.

6.1 Static Networks

Two experiments were performed: the first with two magnetic sensors on the palm, and the second with one flexure sensor and one magnetic sensor on the ankle and near the shoulder respectively. The goal was to map these input values to Cartesian and Spherical coordinates. A total of 25 points were used as samples for network training.

Satisfactory behavior is observed for a two-layer network architecture with a goal of 0.01 and network parameters are exported to Pd MTX files.

The network realization using the abstractions discussed in section 4 is shown in Figure 6. Sensor inputs are routed using the OSC protocol and from the wireless Kroonde interface. The sensor data is preprocessed, calibrated and fed into the network. As is seen in the figure, only one neural network abstraction is presented in the patch which takes care of all the mapping. All the rest are data acquisitions and preprocessing.

Real-time performance is achieved using 200Hz data entrance and the final layer output would correspond to the desired behavior and can be used for further control.

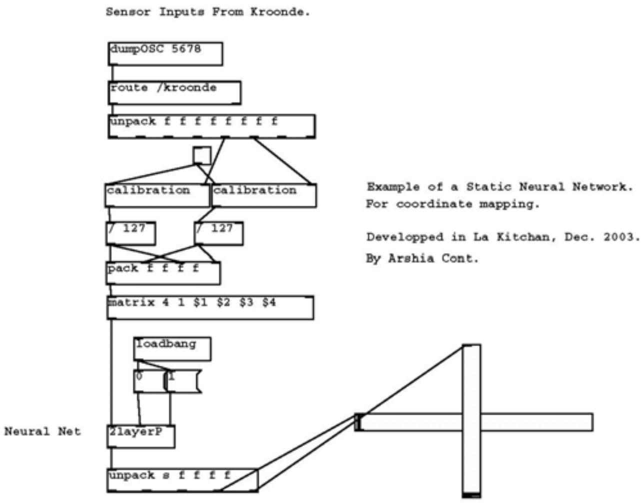

Figure 6. Pd patch for a static network application

6.2 Dynamic Networks

At this stage, our interest is to implement dynamic networks which evolve with time. Gesture recognition is one application. In this manner, we chose to experiment with a circular movement recognition using two accelerometer sensors in two perpendicular axes. For this reason, we use down sampled time delay vectors as entry which introduces a notion of memory for the recognition system.

Sample/target acquisition for training is similar to the previous section but in most applications like this, the target should be assigned manually. We used the Gaussian approach suggested by Fels and Hinton [1] in assigning the target values for each recorded gesture. The training, although heavier than before, converges for more number of neurons and more tries are necessary to meet satisfactory generalization behavior. Figure 7 shows the Pure Data realization of this gesture recognition system.

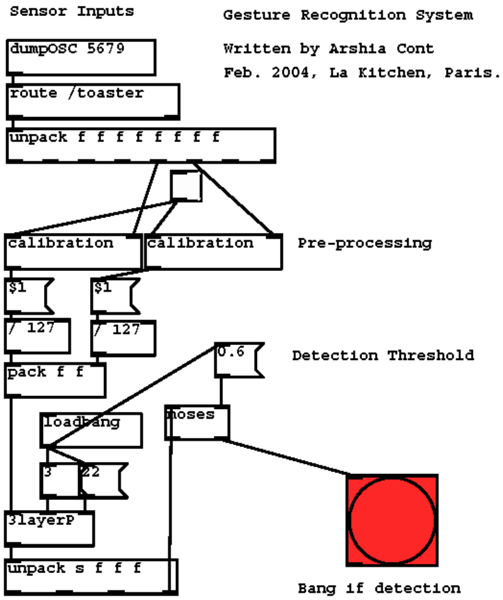

Figure 7. A Gesture Mapping sample network in Pd

The output is presented as a 'bang', that is a bang is observed once the gesture is recognized and a threshold, set and observed during training, is used for this recognition. The same system can be used for more gestures (more bangs) but needs heavier training.

As before, real-time performance is achieved and the network can be used beside other tasks for control of events and high-level parameters. One important aspect of this experience is the generalization behavior of the system. We trained the system with 6 constant speed circles and the network response to a large variety of circles (different speeds and sizes) is satisfactory.

7. FUTURE WORK

Still the training is being done in a Neural Network simulator. One main goal of this project is to construct a training interface in Pd environment with the same design principle shown above.

When using several gestures at the same time, the training becomes very heavy. Although this is a trade-off of the high-level accessibility, we will consider parallel approaches with neural networks for this application in the future.

8. CONCLUSIONS

A general Neural Network mapping scheme was suggested for musical applications due to several reasons, notably ability to generalize the knowledge and its easy and friendly interface. Neural Network abstractions are implemented in the Pd environment using very few externals, in a way that would minimize sophistication of dealing with unnecessary network parameters and complications. Following this, two main applications have been developed and tested using high-speed and high-resolution sensor interfaces, achieving real-time application.

This work finds applications in live and interactive performances and live control of events using a sensor network. Eventually, the user will be able to define and construct a desired mapping to high-level parameters.

9. REFERENCES

[1] Fels, S. S. and Hinton, G. E. Glove-Talk: A neural network interface between a data-glove and a speech synthesizer. *IEEE Trans. On Neural Networks,* vol. 4, No. 1, 1993.

[2] Lee, M., Freed, A., Wessel, D. "Real-Time Neural Network Processing of Gestural and Acoustic Signals", *Proceedings of the 17th International Computer Music Conference, Montreal,* 1991.

[3] Coduys, T. and Henry, C. " Nouveaux matériels de captation haute précision ", *Proceedings of 10th 'Journée d'Informatique Musicale'*, Montbeliard, France, 2003.

[4] Wright, M., Freed, A., " Open SoundControl: A New Protocol for Communicating with Sound Synthesizers", *ICMC* , Thessaloniki, Greece, 1997.

[5] Puckette, M. "Pure Data." *Proceedings, International Computer Music Conference*. San Francisco, 1996.

http://www-crca.ucsd.edu/~msp/software.html

[6] Hagan, M. T., and M. Menhaj, "Training feedforward networks with the Marquardt algorithm," *IEEE Transactions on Neural Networks* , vol. 5, no. 6, pp. 989-993, 1994.