A Type of Radial Basis Function Technique for Control and Time Series Prediction of Positioning Systems

MICHAIL G. PAPOUTSIDAKIS
Dept. of Automation
Technological Institute of Piraeus, Greece
P.Ralli & Thivon 250, Athens, 12244
GREECE
mipapou@teipir.gr http://islab.teipir.gr/web/people.htm

ANTHONY G. PIPE
Bristol Robotics Laboratory
University of the West of England
Coldharbour Lane, Bristol BS16 1QY
UK

Abstract: - It is well known that pneumatic positioning systems are still irreplaceable in many application fields like industrial automation. The maintenance cost, the low level pollution and the high speed of operation, force the use of such systems. The pneumatic piston position control has always been a challenge and engineers have applied many control methods in order to achieve position accuracy. Apart from air compressibility, the most important issue to be solved is the highly nonlinear phenomena inside the cylinder body that become unpredictable over time and long term operations of the system. Multiple friction forces, energy losses and sealing deformations are always present in this type of actuating process. In this research work, an intelligent control approach is implemented for the task, in an attempt to overcome the classical control methods inefficiency. A subcategory method of artificial neural networks is adopted for investigation, which is described in details. All experimentation results, system performance behaviour discussion and possible further improvements, form the rest of this paper body.

Key-Words: - Artificial Neural Networks, Nonlinear Control, Time Varying Positioning System

1 Introduction

The power and usefulness of artificial neural networks have been demonstrated in several applications including speech synthesis, diagnostic problems, medicine, business and finance, robotic control, signal processing, computer vision and many other problems that fall under the category of pattern recognition. For some application areas, neural models show promise in achieving human-like performance over more traditional artificial intelligence techniques. Neural networks (NN) have been shown to be particularly useful in solving problems where traditional artificial intelligence techniques involving symbolic methods have failed or proved inefficient. Such networks have shown promise when applied to problems involving low-level tasks that are computationally intensive, including industrial control. Neural networks, with their massive parallelism, can provide the computing power needed for these problems, though these requirements can make them demanding algorithms when implemented on conventional single-thread execution computer architectures, as is the case here. A major shortcoming of neural networks lies in the long training times that they require, particularly when many layers of weighted connections between neurons are used. Hardware advances should diminish these limitations, and neural-network-based systems are likely to become greater complements to conventional computing systems. Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyse. This expert can then be used to provide projections given new situations of interest and answer "what if" questions. Other advantages of NNs include the *adaptive learning* ability to learn how to do tasks based on the data given for training or initial experience and the s*elf organisation* ability to create its own organisation or representation of the information it receives during learning time. In

addition to this, the *real time operation* characteristic of ANNs allows computations to be carried out in parallel, hence special hardware devices have been designed and manufactured which take advantage of this capability.

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is typically composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. A neural network is an interconnected group of artificial neurons that uses a mathematical or computational model for information processing based in a connectionistic approach to computation. In most cases an ANN is an adaptive system that changes its parameters and/or structure, based on external or internal information that flows through the network. The most common type, the 'Multi-Layer Perceptron (MLP) is shown in figure 1:

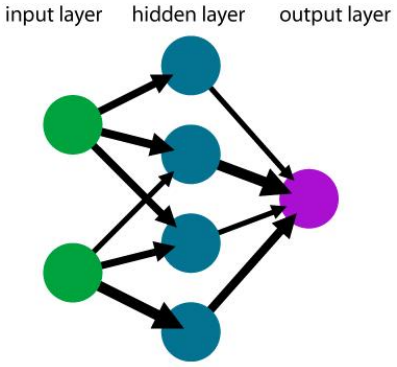

Fig.1 A simple MLP neural network

The above figure represents this type of ANN, which consists of three groups, or layers, of units: a layer of "input" units is connected to a layer of "hidden" units, which is connected to a layer of "output" units. The activity of the input units represents the raw information that is fed into the network. The activity of each hidden units is determined by the activities of the input units and the weights on the connection between the input and the hidden units. The behaviour of the output units depends on the activity of the hidden units and the weights between the hidden and the output units. This type of network is interesting because the hidden units are free to construct their own representations of the input. The weights between the input and hidden units determine when each hidden unit is active and so, by modifying these weights, a hidden unit can choose what it represents. In more practical terms neural networks are non-linear statistical data modelling or decision making tools. They can be used to model complex relationships between inputs and outputs or to find patterns in data. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. This is true of ANNs as well, as in [1], [2], [3], [4] [5], [6], [7], [8], [9] and [10] . An artificial neuron is a device with many inputs and one output. A neuron forming part of a supervised ANN has two modes of operation; the training mode and the using mode. In the training mode, the neuron can be trained to track a particular input-output relationship or set of such relationships. Another categorisation that can be applied here is whether changes from learning are applied after presentation of each input-output training example, or if the errors are summed and used for training only after the ANN has been exposed to the whole set training set. The former is often called 'incremental mode' training, whilst the latter is often called 'batch mode' training. The former, however, allows for 'on-line' training, i.e., ANNs that learn new input-output relationships whilst they are active, as is the case here.

Every neural network possesses knowledge that is contained in the values of the connections weights. Modifying the knowledge stored in the network as a function of experience implies a *learning rule* for changing the values of the weights. All learning methods used for adaptive neural networks can be classified into three major categories, each corresponding to a particular abstract learning task. Usually any given type of network architecture can be employed in any of those tasks. The learning categories are, reinforcement learning, unsupervised learning and supervised learning which was adopted in this paper. Supervised learning incorporates an external teacher, so that each output unit is told what its desired response to input signals ought to be. During the learning process global information may be required. An important issue concerning supervised learning is the problem of error convergence, i.e., the minimization of error between the desired and computed unit values. The aim is to determine a set of weights, which minimizes the error. One well-known method, which is common to many learning paradigms, is the least mean square (LMS) convergence. In supervised learning, a researcher is given a set of examples pairs

$(x, y), x \in X, y \in Y$ and the aim is to find a function f, which is considered as the networks output, in the allowed class of functions that matches the examples. In other words, the goal is to infer how the mapping implied by the data and the cost function, is related to the mismatch between the mapping and the data. It must be stated that a neural network learns off-line if the learning phase and the operation phase are distinct. A neural network learns on-line if it learns and operates at the same time. Usually, supervised learning is performed off-line, whereas unsupervised learning is performed on-line, as in [11], [12], [13], [14] and [15]. Some earlier research work has been carried out in the area of pneumatic systems control by using some kinds of neural network approach. In addition to this, other attempts to apply a neural network controller in a pneumatic system for position control have shown that, although successful performance was indicated, there was still space for further improvement of the controllers, as in [16], [17], [18] [19], [20] and [21].

2 The ANN Type of Control

2.1 Experimental Rig Details

The equipment which was used to undertake the control tasks of this research work, consists of a double acting cylinder (FESTO DSW-32-80-A) of 80mm piston stroke, a proportional servo valve (FESTO MPYE-5-1/8-) with 50msec response time, a linear variable differential transducer, (RDP ACT1000C) as a position sensor for feedback signaling and all necessary gear such as tubing, air preparation and mountings. The pneumatic cylinder is placed vertically because during experimentation its performance was also investigated under variable loads or multiple constant loads. As an interface between the experimental rig and the user, a common PC was used thus all data were recorded into it. A 32-bit microcontroller board was used additionally to the PC in order to compile and execute the ANN control algorithm in the appropriate C programming language code. The microcontroller's output, via wiring, was straight connected to the servo valve, which is the 'controllable' device of the system. The input signal of the microcontroller is the feedback of the position sensor and the target position that the user demands. The detailed block diagram of the pneumatic positioning system is provided in the following figure:

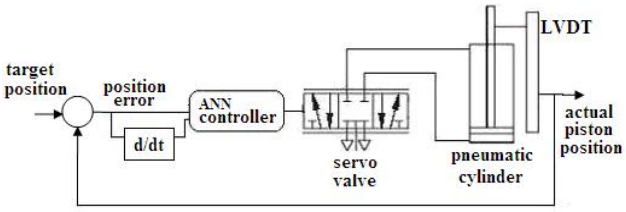

Fig.2 The overall system block diagram

2.2 Radial Basis Function Technique

In this research project it was decided to implement a specific neural network method, which has been used before in nonlinear control tasks, but has never been applied so far in this kind of pneumatic actuator control. This method is called "Radial Basis Function" (RBF) and is a sub-category of the supervised learning neural networks control approach, discussed earlier. The motivation to use this method, apart from the fact that it has not been used before in such pneumatic systems, is that it looks very promising for tracking position control tasks in other related applications, such as electric-motor actuated robot manipulators (see [11], [12] and [13]). The RBF network method is very useful for function approximation, classification and modelling of dynamic systems and time series prediction. They typically consist of three layers: an input layer, a hidden layer with a nonlinear RBF activation function (the neurons), and a linear output layer, shown in the next figure, where x_1, x_2, x_n are multiple inputs and $\hat{y}$ is the single output:

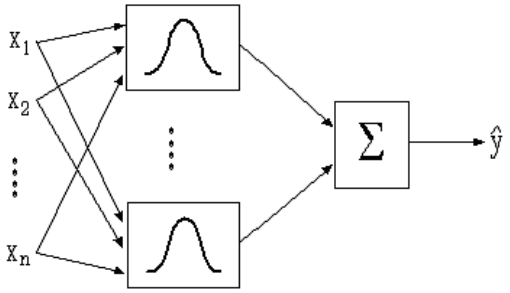

Fig.3 A typically Radial Basis Function network

For the pneumatic system used in this project, the plant has one output, which is the cylinder piston position. At the input of each neuron, the distance between the neuron centre and the input vector is calculated. The output of the neuron is then formed by applying the basis function to this distance. The RBF network output is formed by a weighted sum of the neuron outputs. A bias signal can then be added, but is not used here. For a given input value (or values if there is more than one input), typically

only a small group of neurons is active. As the input value(s) change, so does this group. This is very different from the most common type, the Multi-Layer Perceptron (MLP) where, all neurons are active in forming the output of the ANN no matter what the input value is. The arrows in the figure symbolize parameters in the network. Figure 2 illustrates an RBF network, which is often complemented with a linear part. This corresponds to additional direct connections from the inputs to the output neuron. Mathematically, the RBF network, including a linear part, produces an output given by:

$$g(\theta, x) = \sum_{i=1}^{nb} w_i^2 e^{-\lambda_i^2 (x - w_i^1)^2} + w_{nb+1}^2 + x_1 x_1 + \ldots + x_n x_n \quad (1)$$

Where,

nb is the number of neurons, each containing a basis function. The parameters of the RBF network consist of the positions of the basis functions w_i^1, the inverse of the width of the basis functions λ_i, the weights in output sum w_i^2, and the parameters of the linear part $x_1,\ldots,x_n$. In some cases of function approximation, it is advantageous to retain the additional linear part, but it can also be excluded in many other cases. The parameters are often lumped together in a common variable θ to make the notation compact. The generic description $g(\theta, x)$ of the neural network model can be used, where *g* is the network function and *x* is the input to the network. In the training process, the parameters of the network are tuned so that the training data fit the network model, equation (1), as well as possible. Commonly Gaussian curves are used for RBFs and are chosen for single input, single output control tasks. In this case the basis functions look like "bell"-shapes distributed around the xy-surface. Basis functions are normally of equal size and shape and normally of even separation and hence overlap. Parameters that are open to design decisions include RBF function shape, learning rate and RBF width and separation. Once these parameters are fixed for a given network architecture, the learning algorithm operates on the "weight vector" to gain to obtain the desired performance. An alternative, and complementary, view of how these networks operate is worth noting here, because it may help the reader to understand why some of the work described below and the future enhancements proposed could improve performance. Effectively, this type of ANN is a kind of 'fuzzy look-up table', where nonlinear functions that have overlapping but only local influence with respect to the input values, are weighted and then summed to give an overall output. This means that RBF function width and separation affect the manner in which the network can track important input changes. If the functions are wide, but the input values change rapidly, then it is likely that the neuron activity will 'filter out' any fast changing input values, and the ANN may not react appropriately. However, having a larger number of smaller width and more closely packed RBF functions will reduce the ability of the ANN to generalize between training set examples because the learning will be more highly localized, and there can be an explosion in the required number of neurons to cover the input space. For this project, there are three important advantages of the RBF that make it more useful network rather than the MLP, like:

- There is only one layer of weighted connections. This means that, for any input-output mapping, there is only a singe weight-vector that solves it. This makes learning fast and reliable.
- Neuron activation and learning is localized in the input space. This means that when a new input-output relationship is learned, it will not affect other learned mappings, as is the case for the globally active MLP. In the case of the MLP, if we used it for on-line incremental learning, then we would have to re-train the ANN with the entire training set acquired so far, plus the new example, a gradually increasing sized training set and hence, an impractical approach.
- The weight vector values are applied to the computational feed forward processing *after* input signals have passed through a non-linear function (the basis function).

Since this is a supervised learning architecture, it is assumed that there is a training set. The training set provides known examples of input-output mappings between which the network will interpolate. The algorithm is an iterative one, as for all neural network learning algorithms, i.e., the training examples are repeatedly presented, each time finding an error between the actual RBF net output and the desired output for a given input value. Each error is used to update the weights of those neurons that were active in producing that output value. A proportion of an error is taken by multiplying it by a

"learning rate". For each neuron, this value is then modified by a "weighting factor" proportional to the activation of that neuron. "Incremental" training is used so that the weighted-error values are used immediately to update the weights.

The task of maintaining position stability over long-term operation of the actuator was the scope of experimentation. It was decided to keep as a base the classical PD control method due to its excellent results recorded in [12] and the fact that the PD controller gains are already estimated from prior experimentation. The basic idea here is to use the fixed gain PD controller, and then 'wrap around it' the RBF ANN, so that it sees the same inputs that the PD controller sees, and then adds its output to the PD controller output at a summing junction. The learning signal for the ANN is the error signal between the desired and actual cylinder piston positions. As the PD controller varies in its accuracy due to the time-variant nature of the plant, then the ANN learns to take actions that correct those inadequacies on-line. Throughout the whole experimentation with the pneumatic system during this research project, it was clearly noticeable that the system changes unpredictably due to its response over long term operation. The goal of overcoming the time variant effects in the system was the reason for designing long term experiments and recording whether or not the behavior was improved by applying the RBF neural network method to the plant. It must be stated here that the RBF control algorithm code, all necessary data acquisition code and all communication code between the microcontroller and the pneumatic valve is available from authors upon request since they could not fit in a short paper publication like this. In an attempt to describe the implementation of the neural network controller to the system, some necessary explanations must be provided as an addition to all the above text. There is the fact that a 'temporal advance' factor needed to be evaluated, which corresponds to the temporal inverse of the delay through the plant, i.e., a signal is applied to the plant and sometime later an error is observed that is the result of applying that signal. This means that, when using an error as a training signal, we need to modify weights in the ANN, which correspond to neurons that were active previously by that fixed 'temporal advance' value. This was evaluated experimentally and, when estimated to be 4ms, gave satisfactory performance. In other words, at a 1ms sampling period, the error signal was used to modify those ANN weights that were active 4 times steps previously because they gave rise to the observed error. To this end, the use of the 'ANN wrapped round' structure allows for the PD controller to force the overall controller output values to be bounded during all experiments.

To save time in some of the following experiments, the PD controller was artificially 'detuned' in order to observe and have the ANN correct effects similar to those observed during long-term experiments. In addition, long-term experiments were carried out, turning the ANN learning on and of manually to show its effects. Although learning could be left on permanently, in any practical industrial context, one would not want to have the complex ANN on-line learning code turned on permanently when it is not needed, and it would be simple to automate an on-line error margin estimator that could autonomously turn ANN learning on and off as appropriate so as to allow the ANN to correct for time-varying inaccuracies as and when required.

3 Experimentation and Results

Figure 4 illustrates the overall functionality of the on-line learning algorithm, the axes of figure 4, like all other figures in this section, show time in seconds on the x-axis and the magnitude of the 'position' plot in sensor count values of the LVDT attached to the piston on the y-axis, as measured by the microcontroller.

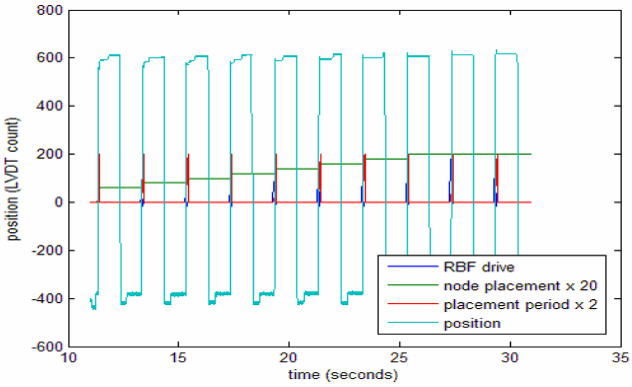

Fig.4 On-line learning functionality

In the above figure, the RBF-node placement algorithm was set to place a maximum of 10 nodes, within a range of time starting from a fixed point relative to the rising edge of the square wave. Node placement was restricted to 10 because, for the medium-power microcontroller used here, this was the maximum number of nodes that could be processed in real-time during the 1ms sampling

period. The decision of whether to place another RBF node or not, is made by the on-line learning algorithm, based on the magnitude of measured error between demanded and actual piston position and whether there is already a node placed within the region. As a result, if the error was larger than a set value and there was no RBF-node centred within a set distance, then a new node was placed. Regardless of whether a new node had been placed or not, if the error was larger than a minimum set value, then a pass of the incremental learning algorithm was then executed. The 'node placement x 20' plot in figure 4 shows the number of nodes currently allocated, and it is easy to see the node count increasing at each of the 'position' plot rising edges. In fact, figure 5 zoomed in at the end of figure 4, shows that 3 nodes were placed during the illustrated 'demand square-wave' period (and the location of placement in the time domain (x-axis position)), bringing the total number of node placed to the maximum of 10 during the period covered by figure 4. The 'node placement x 20' plot was multiplied by 20 to make the values consistent with the y-axis numbered scale. The 'placement period x 2' plot shows the period allowed for RBF-node placement, and can be more clearly seen in figure 5. The triangular shape of the plot has little significance, it is simply the result of plotting the value of an internal variable that is incremented each 1ms sample period after a 'start-time' until it reaches a maximum value of 100 (shown as 200 on the figure due to multiplying this variable by 2 before plotting, again so as to be consistent with the y-axis labelling).

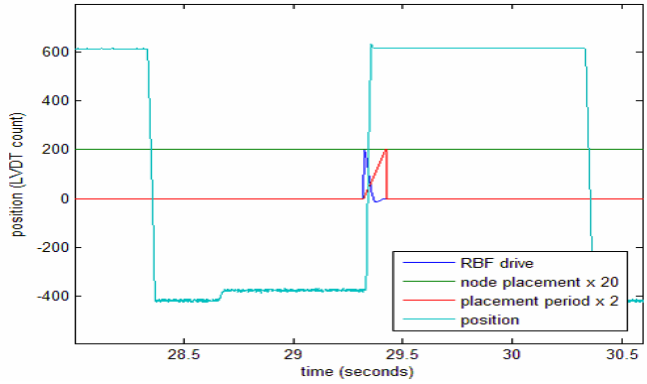

Fig.5 Zoom in at the end of figure 4.

Careful inspection of the 'start time' for RBF-node placement reveals that the time chosen precedes the rising edge of the 'position' plot, illustrating the need to place nodes right at the beginning of the step demand in desired piston position. Indeed, for optimal performance, it might be better to place nodes that predict this demand, though this was not done here, and would only make sense for repetitive cyclic behavior (as actually used here for, though only for test purposes).

In earlier long-term experiments reported in earlier research work as in[22], [23] and [24] it was noted that, given an initially hand-optimized PD-controller, and with fixed PD gains thereafter, the plant response would become progressively over-damped, a situation that could be improved by re-tuning to higher gain values. To simulate the main time-variant effects of these long-term experiments, here, the PD-gains were deliberately de-tuned to give over-damped performance from the outset. Since there is a concern here of showing the *potential* of RBF RAN algorithm in this application domain, this method of obviating time-consuming long-term experiments was considered adequate. With the RBF RAN 'strapped round' the PD controller, and with RBF-node placement restricted to the period around the leading edge of the step demand, it should be able to provide a corrective drive to the system so as to reduce the over-damped effects on the leading edge of the plant's response to a positive step demand. The initial over-damped response is illustrated in figure 6, where the step-demand position value is shown by the dotted line. It is clear to see that, although it reduces during the period of the positive part of the square wave position demand cycle, the plant has a significant error during the steady-state time, in addition to the poor leading edge rise-time. With the learning switched on, after approximately 15 seconds of operation, it can be clearly seen in figure 6 that the rise-time has been improved and steady-state error reduced by the RBF drive signal.

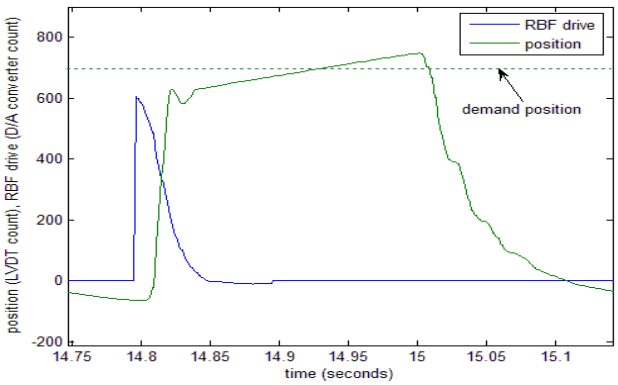

Fig.6 Learning switched on at start of operation

After further operation with the learning switched on, it can be seen in figure 7 that performance is improved further.

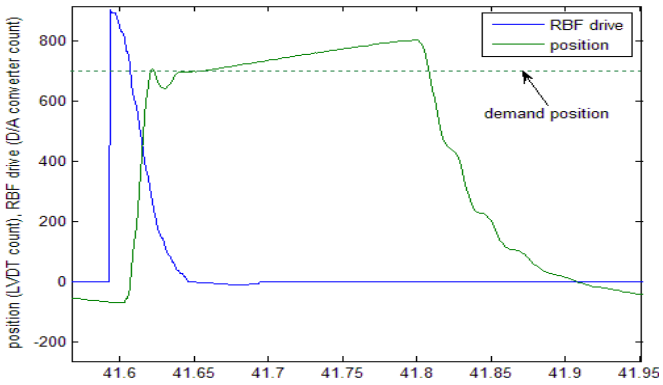

Fig.7 Learning switched on at the middle of operation

The following observation should be noted. A limit had to be placed on the maximum numbers of RBF-nodes (10 in this case, as previously mentioned), and a reasonable limit had also to be set on the maximum size of a node (i.e., its spread or width on the time-domain x-axis of these figures) so that the effect of a given node was appropriately local in the time-domain.

In order to illustrate the behaviour and the stability of the control algorithm another challenging experiment was designed. In order to simulate a demanding industrial environment, the effects of long-term operation were investigated using a different cylinder to the one used for tests earlier here. A different cylinder (although the exact same FESTO model) was used in order to illustrate the versatility of this algorithm, as well as indicating minor limitations to the RBF RAN algorithm as it currently stands. This new cylinder displayed different long-term time-variant characteristics to the first one. Over extended periods of operation, instead of requiring *increase* in both P and D gains, manual re-tuning resulted in a requirement to *reduce* the derivative (D) gain to ameliorate an oscillatory overshoot in the rising edge response to a step position demand. The details of this difference in long-term characteristics is not fully understood within the scope of this research work, but it can easily be imagined that the balance between the plethora of components affected by the passage of time during an experiment and, particularly, the concomitant temperature effects, could be different from one cylinder to another even though they might have been completely identical at manufacture. These differences were compounded, in this case, by the fact that the new cylinder was, literally, newer, and had been subjected to far fewer previous operations during its lifetime. Whilst the initial response of this cylinder to a step demand input is plotted in figure 8, after approximately 37 minutes of continuous operation in this square wave demand regime, the plant response appeared as shown in figure 8.

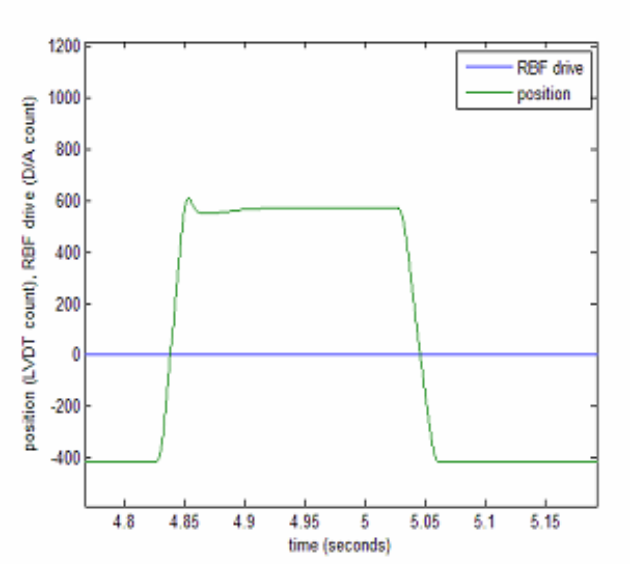

Fig.8 Initial response to a step input

The increased leading edge oscillatory response is clearly visible in figure 9.

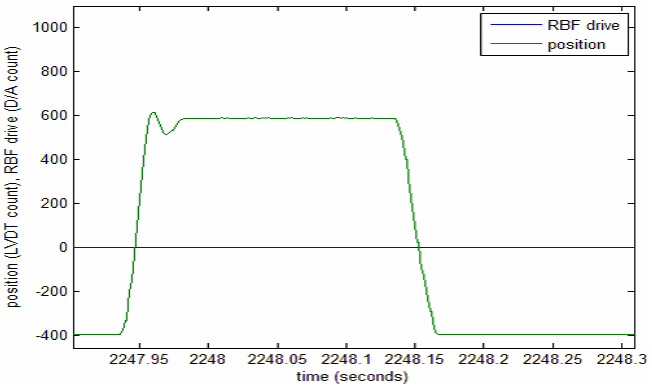

Fig.9 Long-term response to a step input without RBF

To reduce this effect without having to manually re-tune the PD controller, the RBF RAN was, again, strapped round it. After some additional time a useful, if incomplete, reduction in oscillatory response was achieved, as can be seen in figure 10, thus proving once again the beneficiary affect of the ANN method adopted in the pneumatic system.

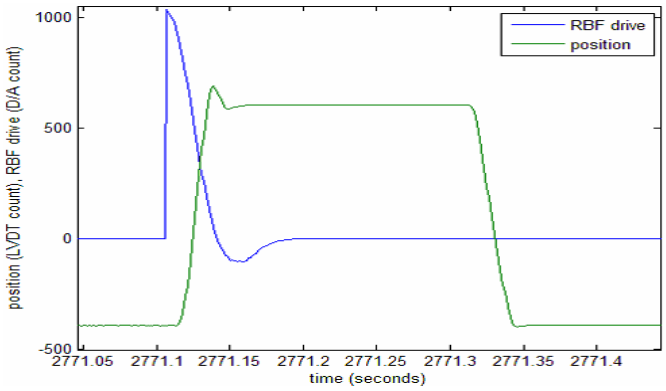

Fig.10 Long-term response to a step input with RBF on

However, it should be noted here, that to allow the RBF ANN to respond appropriately to this localised effect, the size (i.e., spread or width) of the RBF-nodes had to be reduced below that used in the first experiment above. Although this is not a problem in itself, it prompts the notion that a mature version of this algorithm would not only be able to place many more neurons across the temporal domain, but also be able to change the size of, and distance between, neurons dynamically to suit a given error scenario that it is meant to have an effect on. These modifications would be required for an 'industry ready' solution, but are beyond the scope of this research work. Indeed, it can be seen from figure 10, that the '10-neuron limit' already restricts the appropriateness of the RBF drive signal in attempting to correct for the oscillatory leading edge error; its response is too coarsely grained for the frequency of oscillation. What is really needed is an array of smaller and closer packed RBF-nodes in the region immediately preceding the oscillatory response but when this was attempted, 10 neurons was simply an insufficient number to cover the whole area of concern in the leading edge of the waveform.

4 Conclusion

A pneumatic position system is considered to be a critical piece of equipment in various fields of application. In modern industry, pneumatics has a significant role in many robotic applications, despite the high complexity and the nonlinearities that these systems introduce. In this research project, an overall understanding of the dynamics of such a system during execution of the task of position control was developed. At the same time, a significant "branch" of the modern intelligent control methods, the on-line learning neural network control, was applied to a pneumatic positioning system. As a quick review of ANN implementation in a real pneumatic positioning rig, it can be recorded that the power of this approach, for correcting the long-term time variant characteristics of this plant, is highlighted in this paper. The stability of the system is ensured, and there is no need for manual re-tuning of the controller over long-term operation, like classical control approaches. The purely 'measured error-based' approach adopted in the on-line learning Radial Basis Function method would not suffer from any of problems like re-tuning and would seem, therefore, to be the approach holding the most promise. With all the above details of the control methods applied to the system, the basic conclusion is that the RBF Neural network control approaches form a robust set of solutions that conform well to the aims of this research project. This last outcome means that the abstract hypothesis of whether or not this method would be beneficiary to the system is upheld.

5 Further Research Work

A successfully controlled pneumatic actuator could be an excellent part of any mechanical system as a force generator or movement and transport element. A deeper investigation of the dynamics of the system, incorporating the effects of temperature and energy losses, would allow more precise data to be included as separated parameters in the control algorithm. It was verified experimentally that internal time-variance due to temperature changes and load influences, are major factors that must be taken fully into account in devising a robust controller, and it would be beneficial to include them as inputs to the system. Depending on the requirements of the application of the pneumatic actuator, another controller input could be air pressure changes, detected by using appropriate air pressure sensors. These additional parameters would help the researcher to design a robust and variably compliant positioning system, providing a wider range of applications. However, perhaps even more worthy of attention is the control of a pneumatic system with more than one actuator operating together, at the same time. In fact, it is intended that this is where this research will finally lead, as an extension to this paper work. The development of a combinational actuator circuit, which will control all degrees of freedom of a robotic arm, is the final and most challenging further research work that, it is hoped, will be undertaken in the near future.

References:

[1] C.H Chen, "Fuzzy Logic and Neural Network Handbook", *Mc Graw-Hill Education*, 1996.

[2] Juan R. Rabunal, Julian Dorado, "Artificial Neural Networks in Real Life Applications", *Idea Group Publishing*, 2006.

[3] Libor Pekar and Filippo Neri, "An Introduction to the Special Issue on Advanced Control Methods: Theory and Application", WSEAS Transactions on Systems, issue 6, vol. 12, June 2013

[4] L.P.J. Veelenturf, "Analysis and Applications of Artificial Neural Networks" *Prentice Hall International*, 1995.

[5] Michael A. Arbib, "The Handbook of Brain Theory and Neural Networks", *THE MIT PRESS*, Second Edition, 2003.

[6] Neri F., "Empirical Investigation of Word-Of-Mouth phenomena in Markets: a Software Agent Approach", WSEAS Transactions on Computers, WSEAS Press (Wisconsin, USA), issue 8, vol.4, pp.487-994

[7] Lubomir Macku and David Samek, "Two Step, PID And Model Predictive Control Using Artificial Neural Network Applied On Semi-Batch Reactor" WSEAS Transactions on Systems, issue 10, vol.9, October 2010

[8] Mousa AL-Akhras and Maha Saadeh, "Automatic Valuation of Jordanian Estates Using a Genetically-Optimised Artificial Neural Network Approach" WSEAS Transactions on Systems, issue 8, vol.9, August 2010

[9] Yunquan Ke and Chunfang Miao, "Stability Analysis of BAM Neural Networks with Inertial Term and Time Delay", WSEAS Transactions on Systems, issue 12, vol.10, December 2011

[10] Hassen Mekki and Mohamed Chtourou, "Variable Structure Neural Networks for Real Time Approximation of Continuous Time Dynamical Systems Using Evolutionary Artificial Fields", WSEAS Transactions on Systems, issue 2, vol.11, February 2012

[11] Martin Anthony, Peter Bartlett, "Neural Network Learning: Theoretical Foundations", *Cambridge University Press*, 1999

[12] Alexander Lenz, Anthony G. Pipe, "A Dynamically Sized Radial Basis Function Neural Network for Joint Control of a PUMA 500 Manipulator", *IEEE International Symposium of Intelligent Control*, Houston, Texas, October 5-8, 2003

[13] M. Papoutsidakis, A.Pipe and G.Chamilothoris, "Supervised Learning Method of Neural Networks in a Non-linear and Time Depended Control Process", 21st Mediterranean Conference on Control and Automation, June 2013, Chania, Greece

[14] Jin Y, Pipe A.G., Winfield A., "Stable Neural Network Control for Manipulators", *IEEE Journal of Intelligent Systems Engineering*, 1993, pp213-222.

[15] Lewis F.W., Jagannathan S., Yesildirek A, 1999, "Neural Network Control of Robot Manipulators and Non-linear Systems", *Taylor & Francis Books*.

[16] Wang Qi Tao Qian Hou Linqi Cai Hegao, "On-line Learning Neural Network Controller for Pneumatic Robot Position Control", *0-7803-4778-1, IEEE*, 1998.

[17] Kale Harbick, Gaurav S. Sukhatme, "Speed Control of a Pneumatic Monopod using a Neural Network", *Tech. Rep. IRIS-02-41, Institute for Robotics and Intelligent Systems*, USC, 2002.

[18] Gi Sang Choi, Han Koo Lee, Gi Heung Choi, "A Study on Tracking Position Control of Pneumatic Actuators Using Neural Network", *0-7803-4503-7, IEEE*, 1998.

[19] T. Hesselroth, K. Sarkar, P. van der Smagt, K.Schulten, "Neural Network Control of a Pneumatic Robot Arm", *IEEE Transactions on Systems, Man, and Cybernetics*, vol 24, No 1, 1994.

[20] Y.J. Liu, X.Z. Kong, Z.W.Li, "Pneumatic Proportional Position Control System Based on Neural Networks", *Trans, Tech Publications, Switzerland*, 2004.

[21] K.K. Ahn, D.C Thanh, "Nonlinear PID Control to Improve the Control Performance of the Pneumatic Artificial Muscle Manipulator Using Neural Network", *Journal of Mechanical Science and Technology*, vol. 19, No1, pp. 106-115, 2005.

[22] Papoutsidakis M.G., Chamilothoris G.E, F. Dailami, N. Larsen and A.G. Pipe, «Accurate Control of a Pneumatic System Using an Innovative Fuzzy Gain-Scheduling Pattern», *International Conference on System Science and Engineering*, Budapest, Hungary, October 2005.

[23] M. Papoutsidakis, C.Alafodimos and A.Pipe, "A Collaborative PID and Intelligent Control Method for a Nonlinear Positioning System in Simulation", the 2012 IEEE International Conference on Control Applications (CCA), October 2012, Dubrovnik, Croatia

[24] Michail Papoutsidakis, G.Gafas, S.Kanavetas and G.Chamilothoris "Modeling and Simulated Control of Non-linear Switching Actuation Systems", 8th International Conference on System Science and Simulation in Engineering, October 2009, Genoa, Italy