

AN ADAPTIVE NEURAL NETWORK BASED CONTROLLER FOR MACHINE TUNING OF ELECTRONIC CIRCUIT BOARDS.

This invention relates to an auto-tuning controller for tuning electronic circuit boards, for which the relationship governing the control parameter(s) and the controlled variable(s) is time invariant during the course of tuning. Although the invention is directed towards boards with a stationary characteristic, this does not preclude the deployment of this invention to control a plurality of boards, in which the boards in question share significant degree of common behaviour. The task of tuning electronic circuit boards in a mass production environment represents a suitable candidate for application of this invention. In this task domain, accurate tuning must be accomplished within a short time interval. The difficulty is further compounded by variations introduced during the manufacturing process. A preferred embodiment of the invention describes a system that adjusts the brightness, contrast, tint and color settings of TV's chassis boards by the manipulation of a set of associated registers.

BACKGROUND AND PRIOR ART

In the manufacturing industry, the tuning of electronic circuits is still largely performed by experienced human operators. When multiple control items are involved, the tuning process usually proceeds in a sequential manner. The order of adjustment of the various control items to be adjusted is virtually determined by human perception, and the magnitude of each adjustment is a rough approximation of the deviation involved. A system to dynamically improve TV picture qualities that used a combination of fuzzy rule based reasoning and neural network techniques has been described in U.S. Pat No. 5,327,228. In the preferred embodiment of that patent, the distribution of facial and non-facial tones in a TV picture are analysed. The necessary control is then accomplished through the use of neural network techniques coupled with a set of pre-determined fuzzy reasoning rules.

Akio Ukita, et. al. described a method to tune microwave circuit boards in the article "Automated Tuning of an Electronic Circuit Board Using the Artificial Neural Network Approach", Journal of Intelligent Manufacturing Systems, Chapman & Hall, 1996. The adjustment process is divided into two distinct portions namely trimmer selection and trimmer rotation. Separate neural networks are assigned to handle the respective tasks.

The capability of artificial neural networks as a powerful tool for function approximation has been well documented and is known to those in the information processing art. References such as "Artificial Neural Networks: Approximations and Learning Theory", White, Blackwell, 1992 and "Multi-Layer Feedforward Network Are Universal Approximators", Hornik, Stinchcombe and White, Neural Network, Volume 2, pp 359-399, 1989 proved that feedforward neural networks with sigmoidal activation functions in the hidden layers are universal function approximators. Recently, Radial Basis Function networks have also been shown to be capable of exhibiting desirable properties just like feedforward networks with sigmoidal activation functions in the hidden layers. A notable article is "Multivariable Functional Interpolation and Adaptive Networks", Broomhead and Lowe. Complex Systems, Vol. 2, pp 321-355, 1988. The application of neural networks for control problems can also be found in literature like "Neural Networks for Control", Miller et. al., MIT Press, 1990 and "A Multi-Layer Neural Network Controller", Psaltis et. al., IEEE Control Systems Magazine, pp 17-21, April 1988.

PROBLEMS THAT THE INVENTION IS TO SOLVE

The main object of this invention is to provide a means for automated, fast, accurate and reliable tuning of multiple electronic circuit boards with time-invariant characteristics. In contrast to manual tuning methods, the proposed method tunes multiple control parameters in parallel. Tuning accuracy is enhanced by employing powerful neural network techniques instead of relying on imprecise human adjustments. The electronic circuit boards involved are expected to share some basic common characteristics, but can, and normally do, differ from one another due to idiosyncratic deviations.

Sequential tuning of multiple control parameters is both inefficient and slow when a strong cross-correlation exists between the control items. Altering the value of a particular controlled variable by simply tuning its corresponding control parameter will not only change the intended controlled variable, but other variables will also be affected accordingly. Thus a natural solution is to adjust all control parameters in parallel, which is the inherent mode of operation in neural networks. Moreover, a substantial amount of time can be saved by tuning in parallel.

Inconsistency in production control and shifts in component values are inadvertent occurrences in manufacturing industry. These factors contribute to considerable deviations commonly found in electronic circuit boards of the same model. Controllers which are designed using conventional rule reasoning, are not flexible and cannot cater for such discrepancies. Even neural network based solutions that are trained offline will be inappropriate since they are not adapted to changes in the characteristics of the electronic circuit boards. An on-line adaptation mechanism to be described herein attempts to address this condition through efficient management of training data.

Using neural network techniques, an electronic circuit board with a non-linear input-output relationship can be controlled. Requirements of linearity, or piecewise linearity, at the region of operation need not be obeyed. The neural network based controller is trained to model the input-output mapping over the extent of the entire space, thus allowing any arbitrary starting values for the control parameters.

SUMMARY OF THE INVENTION

The application of neural network techniques endows this invention with the following desirable attributes:

1. The capability to control electronic circuit board(s) with non-linear input-output characteristics.
2. The ability to adapt with moderate shifts in electronic circuit boards' behaviour, which is useful when a plurality of boards are to be adjusted to an identical desired response.
3. The fast, simultaneous tuning of a plurality of control parameters as a result of inherent parallel execution of the neural network algorithm. Using Conjugate-Gradient weight optimization, the training process is shown to converge in a shorter time than the standard Backpropagation algorithm as outlined in "Restart Procedures for Conjugate Gradient Method", Powell, Mathematical Programming, Vol. 12, pp 241-254, 1977.

To achieve the goal of on-line adaptation, an efficient data management method for training pairs is required to maintain superior generalization in the neural network based controller. This is significant because live streams of data are presented to the

controller during the tuning process. A novel method is adopted for this cause. Basically, the in-coming data is stored and manipulated to produce new training pairs for learning in the neural network controller. The training set is then updated selectively by virtue of the regions in the input space occupied by the new training pairs.

More specifically, in accordance with the present invention there is disclosed 1 an on-line adaptive controller for sequentially tuning a plurality of electronic circuit boards, said controller comprising:

(a) A reference signal generator for generating a desired response for at least one variable to be controlled;

(b) an error signal generator connected to said reference signal generator and the circuit board to be tuned and arranged to output an error signal indicative of the difference between said desired response and the actual value of said variable(s) to be controlled;

(c) an adjustment calculator including a neural network, said calculator being connected to said error signal generator, and arranged to output a weighted control signal to said circuit board in response to the error signal and the current value of the control signal;

(d) a training pair manager connected to said circuit board and said adjustment calculator to receive said weighted control signal and the actual value of said variable(s) to be controlled, and outputting to a neural network based learning network a corresponding updated training set including new and aggregated training pairs;

(e) said neural network based learning network being independent of, but having substantially identical architecture to, the neural network of said adjustment calculator and being connected to said adjustment calculator to determine the transfer function of its neural network by updating its internal weight parameters; and

(f) synchronising means interconnecting said learning network and said adjustment calculator to synchronise said weight parameter updating.

Furthermore, there is also disclosed a method of sequentially tuning a plurality of electronic circuit boards, said method comprising the steps of:

(a) calculating an adjustable weighted control signal based upon a generated error between a reference signal and an actual value of at least one variable to be controlled;

(b) using a first neural network in the calculation of said control circuit;



(c) using an independent, but substantially identical neural network based learning network to vary the weighting of said control signal; and;

(d) training said learning network with the weighted control signal and the actual value of said variable(s) to be controlled to adaptively vary said control signal weighting.

BRIEF DESCRIPTION OF THE DRAWINGS

An embodiment of the present invention will now be described with reference to the drawings in which:

FIG. 1 is a block diagram showing the generic configuration of the adaptive neural network based controller for tuning of a controlled electronic circuit board 10.

FIG. 2 is a flowchart describing the operations of the neural network controller in a typical control process where electronic circuit boards are to be tuned.

FIG. 3 shows graphically the "bins" updating mechanism employed in the training pair manager 38 depicted in FIG. 1.

FIG. 4 is a block diagram showing the preferred embodiment for the invention.

FIG. 5 shows three voltage waveforms measured from the TV chassis board 10 and the interpretation of these waveforms to form four TV signal readings: Brightness, Contrast, Tint and Color.

FIG. 6 shows the implementation of the signal offseer 14.

FIG. 7 shows the implementation of the normalizer 1 15, which normalizes the error signal $e(k)$ and register values $u(k)$.

FIG. 8 illustrates the feedforward, layered structure of the neural network utilized for the implementation of the adjustment network 16 and learning network 35.

FIG. 9 shows the implementation of the denormalizer 17.

FIG. 10 shows the implementation of the register offseer 18.

FIG. 11 describes the implementation of the set-point database 32 using a dynamic list to store observation pairs $(u(k), y(k))$ acquired from measurements in TV chassis board 10.

FIG. 12 shows the interaction between the set-point database 32 and training pair generator 33. New training pairs are created by computing the vector difference between the latest observation pair and the contents in the dynamic list of the set-point database 32.

FIG. 13 shows the implementation of the normalizer 2 designated 34.

This Sheet Has Been Replaced



THIS SHEET HAS BEEN
REPLACED

FIG. 14 presents the timing diagrams explaining the role of a RDY signal for the synchronization of the weight parameters between $W1'$ and $W2'$ in the learning network 35 and $W1$ and $W2$ in the adjustment network 16.

FIG. 15 is a flowchart describing the operations of the individual components within the adjustment controller 20 during an typical adjustment cycle for a TV chassis board 10 in the preferred embodiment; and

FIG. 16 is a flowchart describing the operations of the individual components within the adjustment learner 40 during a typical learning cycle for a TV chassis board 10 in the preferred embodiment.

DETAILED DESCRIPTION

Described below with reference to Fig. 1 is the typical operation of the neural network based controller in the generic configuration. An adjustment controller 20 and its related components provide appropriate control signal(s) to a controlled electronic circuit board 10 such that a user-specified desired response can be achieved. The controlled electronic circuit board 10 in Fig. 1 encompasses relevant measurement modules required for actual physical realization of this embodiment. As the output of $y(k+1)$ the controlled board 10 cannot be *exactly* equivalent to the reference signal y° produced by a reference signal generator, a pragmatic approach is to specify a tolerance level for the controller 20 to achieve.

In the following paragraphs, the roles of the major components involved will be discussed. Two distinct operating phases can be identified namely an adjustment phase and a training phase. The reference numerals 90 and 91 in Fig. 2 denote the steps involved in the adjustment phase and training phase respectively. Though shown to be operating in sequential manner, these two phases can proceed concurrently and are handled by the adjustment controller 20 and an adjustment learner 40 (Fig. 1) respectively. These are described separately below and the relationship between the components 12, 35 and 38 will also be explained with the aid of Fig. 1.

The controlled electronic circuit board 10, has input and output signals designated as $u(k+1)$ and $y(k+1)$ respectively. The indices k and $k+1$ indicate discrete time steps during tuning and z^{-1} denotes a unit time delay element 55. The bold typeface associated with the signals indicate that they are generally vectorized quantities.

A reference signal generator 60 provides the desired response denoted by y° . This



signal is compared against the current output $y(k)$ of the controlled electronic circuit board 10, their difference produces an error signal $e(k)$:

$$e(k) = y(k) - y^{\circ} \quad (1)$$

During the adjustment phase 90, the adjustment calculator 12 computes the change to the control parameter $\Delta u(k)$, while taking both $e(k)$ and the current value of the control parameter $u(k)$ as inputs. $\Delta u(k)$ refers to the approximate change required for $u(k)$ to reduce $e(k)$. In this implementation, the adjustment calculator 12 preferably houses a layered, feedforward neural network just as the structure described in "Learning Internal Representation by Error Propagation", Rumelhart et. al., in "Parallel Distributed Processing: Explorations in Microstructure of Cognition", MIT Press, 1986. Simple processing elements are arranged in layers, with full inter-connections between two adjacent layers. The computation of $\Delta u(k)$ proceeds quickly since only a forward propagation of the input signals is required.

During this process, the weight parameters of the neural network in the adjustment calculator 12 denoted by W remain fixed. The new value of the control parameter $u(k+1)$ is calculated as :

$$u(k+1) = u(k) - \Delta u(k) \quad (2)$$

The new setting for the control parameter is then imposed onto the controlled electronic circuit board 10 which produces a response $y(k+1)$ accordingly. This completes the adjustment phase 90 in the overall control process. The control process is repeated until the error $e(k)$ falls below the tolerance specified.

The behaviour of the adjustment controller 20 is shaped by an on-line learning strategy. This implies that the adjustment calculator 12 will constantly adapt its internal weight parameters by observing a stream of signals $u(k)$ and $y(k)$. To accomplish this mechanism, a neural network of identical architecture to that of the adjustment calculator 12 is required. This neural network is called the learning network and denoted as 35 in Fig. 1.

During the training phase 91, the learning network 35 retrieves training data from the training pair manager 38. The training pair manager 38 stores the previous history of $u(m)$ and $y(m)$ for $m = \{k-p, k-p+1, \dots, k\}$, where p indicates the depth of memory. When the latest observation pair $(u(k+1), y(k+1))$ is presented to the

controller, the training pair manager 38 computes the difference between $(u(k+1), y(k+1))$ and the p pairs of data stored. This results in the creation of p new supervised training pairs of the format $(u(k), \Delta y(k); \Delta u(k))$ where ; delimits the input signal from the teacher signal.

A novel approach is adopted here to update the training pairs within the training pair manager 38. This method is illustrated in Fig. 3. The rationale behind the adoption of this approach is to ensure uniform approximation in the input space of the learning network 35 and adjustment calculator 12. In this method, the input space formed by a concatenation of signals $u(k), \Delta y(k)$ is divided into discrete, disjoint partitions called "*bins*". All bins house a single training pair, but their size may not be equal as shown in Fig. 3. In Fig. 3, $u(k)$ and $\Delta y(k)$ are represented as uni-variate signals for simplicity of illustration only. In physical applications, they can be multi-dimensional quantities.

At any time step k , new training pairs such as A' , B' and C' will be dynamically created from the subtraction between latest observation pair $(u(k), y(k))$ and existing observation pairs found in the training pair manager 38. If the corresponding bin is occupied by a previous training pair, the new training pair will overwrite it as shown by replacements between A and A' , B and B' . On the other hand, if a new training pair C' is generated with its associated bin unoccupied, it will be directly stored in the database. This mechanism helps to preserve superior approximation by attempting to distribute training pairs evenly, without an exponential increase in the number of training pairs. The distribution of bins over the input space can be determined adaptively through some means. As shown in Fig. 3, the density of distribution decreases with increasing distance from the centre of the input space at $D(0.5, 0.5; 0.5)$. This strategy is to bias the neural networks found in the learning network 35 and adjustment calculator 12 towards better generalisation near the target region, where adjustments have small magnitudes.

Once the updating of the training pair manager 38 is completed, all training pairs existing in occupied bins will be aggregated to form a training set. The resulting training set is then presented to the learning network 35 for Conjugate Gradient training. At the end of the learning process, an active RDY signal is sent to the adjustment calculator 12 signifying that its internal weight parameters W are to be replaced by new values labelled collectively as W' in Fig. 1. As the discrete time step increments from k

to $k+1$, the weight parameters will be transferred from the learning network 35 to the adjustment calculator 12. This completes one cycle of the training phase 91.

A typical operation can be summarized with the aid of a flowchart shown in Fig. 2. Steps 81, 82, 83, 84 and 85 constitute the adjustment phase 90, while steps 86 and 87 are operations pertaining to the training phase 91. Firstly, the discrete time index k is initialized to 0 at step 80. The current response of controlled electronic circuit board 10 is then measured and the deviation from target $e(k)$ calculated at step 81. A decision is made at step 82 to determine whether the error is acceptable. If further adjustments are required, a corrective control signal will be computed using the adjustment calculator 12 in step 83. The adjustment calculator propagates the signals $u(k)$ and $\Delta y(k)$ through its layers and produces $\Delta u(k)$. The new value for control parameter $u(k+1)$ is computed using equation (2) in step 84. The corresponding response of the electronic circuit board 10 $y(k+1)$ is then measured in step 85. At this stage, one cycle of adjustment phase has 90 been completed.

The operations labelled 86 and 87 are parts of the training phase 91. In step 86, new training pairs are created from the training pair manager 38 by computing the difference between the latest observation pair $(u(k+1), y(k+1))$ against previously stored p pairs. All "bins" affected are updated with the new training pairs. Next, the learning network 35 retrieves the training data from the training pair manager 38 and initiates the Conjugate-Gradient training process in step 87. Once training terminates, the signal RDY will be pulled high for a period of time to enable transfer of weight parameters W' from the learning network 38 to replace the corresponding weight parameters W in adjustment calculator 12. The training phase 91 is now completed and the next adjustment cycle can repeat at step 81 after the system increments the time index by 1 in step 89. As mentioned previously, the training phase 91 and adjustment phase 90 can proceed in a concurrent manner, but they are depicted as serial operations in Fig 9.

Fig 4. shows the block diagram of the preferred embodiment of this invention. In this embodiment, the adaptive on-line neural network based controller is applied to dynamically tune three 7-bits and one 8-bits registers, which directly control the settings of four TV signals: Brightness $y_b(k)$, Contrast $y_c(k)$, Tint $y_t(k)$ and Color $y_s(k)$. Collectively, we shall denote these four signals as a vector $y(k)$. As in the previous section, k indexes the discrete time step. The reference numeral 10 represents the controlled electronic circuit board. hereinafter called the TV chassis board.

A practical implementation of this embodiment includes configuring the neural network controller as part of a picture quality adjustment station found along TV production lines. In this scenario, a plurality of TV chassis boards will be sequentially presented to the station for tuning. Since the input-output characteristics of individual chassis board differs due to shifts in component values, the adjustment controller 20 must model the general characteristics of a particular chassis type. By virtue of its superior function approximation capabilities, a neural-network based controller is the most suitable candidate for this task. To capture significant deviations over multiple chassis, the adjustment controller 20 must be adaptable during the tuning process. This requirement is addressed by an on-line learning strategy provided by the adjustment learner 40.

To interface the TV chassis board 10 to the neural network based controller, electronic circuitry is required which is represented by the measurement interface denoted as 8. Fig. 5 shows how the four TV signal settings: brightness $y_b(k)$, contrast $y_c(k)$, tint $y_t(k)$ and color $y_s(k)$ are measured from the TV chassis board 10 using the TV signal digitizer 2. Firstly, a test pattern comprising color bars and rainbow is injected into the TV chassis board 10. By measuring the resulting voltage levels of the TV chassis board 10 at various vertical scan positions, three waveforms are captured. They are shown in Fig. 5. By sampling the waveforms at designated timings and measuring the voltage levels, the TV signal digitizer 2 acquired the readings A, B, C, D and E indicated in Fig. 5. Readings A, B and C correspond to the TV signal settings for brightness $y_b(k)$, contrast $y_c(k)$ and color $y_s(k)$ respectively, while the ratio of D over E gives the tint $y_t(k)$ setting. Examples of values for desired settings and tolerances are included in Fig. 5. Integrated circuits which can perform the functions of TV signal digitizer 2 are commonly used as instrumentation devices and are well known to those skilled in the art. The control parameters $u_b(k)$, $u_c(k)$, $u_t(k)$ and $u_s(k)$ (vectorized as $u(k)$) in this embodiment are data available from EEPROM memory in the TV chassis board 10. Using a micro-controller 7, this piece of information can be transferred to both the adjustment controller 20 and adjustment learner 40 via the Inter-IC (I²C) bus 3. Writing of data to the EEPROM memory can also be accomplished through the interface provided by the I²C bus 3 and micro-controller 7. The working principles of I²C-bus and its protocols are common features in TV receivers, and hence they should be familiar to technicians and engineers skilled in this area.

During a "read" operation from the TV chassis board 10, the value for the four TV signals brightness $y_b(k)$, contrast $y_c(k)$, tint $y_t(k)$, color $y_s(k)$ and the data on I²C bus 3 are transferred to the adjustment controller 20 and adjustment learner 40. This marks the start of a typical adjustment cycle. Firstly, we shall describe the adjustment controller 20 and its components. The function of the signal off-setter 14 is to compute the error signal $e(k)$ by subtracting from $y(k)$ the target y° . The operation performed in 14 is pictured in Fig. 6. In Fig. 7, reference numeral 15 shows a normalizer 1 which scales two inputs $e(k)$ and $u(k)$ to compatible ranges for processing in an adjustment network 16 (Fig. 4).

Fig. 7 illustrates the normalization process, showing signals $e(k)$ and their associated registers $u(k)$. Note that the quantities shown in Fig. 7 are vectors comprising four elements each, the multiplication and addition operations are applied individually to each element. In Fig. 7, $y_{\min}$, $y_{\max}$, $u_{\min}$ and $u_{\max}$ represent the minimum and maximum values of the TV signals, the minimum and maximum values of the registers values respectively. $u_{\min}$ and $u_{\max}$ are determined by the number of bits used to represent the register values, while $y_{\min}$ and $y_{\max}$ are set to the extreme readings possible for the TV signal settings. During system start-up, all 16 quantities are loaded into the memory of normalizer 1 15 by reading from a disk file. The equations governing the normalization process in normalizer 1 15 are set out below:

$$\begin{aligned}
 \hat{e}(k) &= \frac{e(k) - \Delta y_{\min}}{\Delta y_{\max} - \Delta y_{\min}} \\
 &= \frac{e(k) - (y_{\min} - y_{\max})}{(y_{\max} - y_{\min}) - (y_{\min} - y_{\max})} \\
 &= \frac{e(k) - y_{\min} + y_{\max}}{2(y_{\max} - y_{\min})}
 \end{aligned} \tag{3}$$

$$\begin{aligned}
 \hat{u}(k) &= \frac{u(k) - u_{\min}}{u_{\max} - u_{\min}}
 \end{aligned} \tag{4}$$

The outputs of normalizer 1 15 comprise vectors $\hat{e}(k)$ and $\hat{u}(k)$ which become the inputs to the adjustment network 16. The purpose of the adjustment network 16 is to compute a normalized corrective signal $\hat{u}(k)$ to reduce the error $\hat{e}(k)$. No training is

carried out in this network, only forward propagation is necessary to obtain $\hat{u}(k)$. The configuration of the adjustment network 16 is pictured in Fig. 8, which is a three-layer feedforward neural network. The inputs to the network comprise four normalized error signals $\hat{e}_b(k)$, $\hat{e}_c(k)$, $\hat{e}_s(k)$, $\hat{e}_t(k)$ and four normalized register values $\hat{u}_b(k)$, $\hat{u}_c(k)$, $\hat{u}_s(k)$, $\hat{u}_t(k)$. The teachers are four normalized change to the register values $\Delta\hat{u}_b(k)$, $\Delta\hat{u}_c(k)$, $\Delta\hat{u}_s(k)$, $\Delta\hat{u}_t(k)$. Both the hidden and output layer neurons use log-sigmoid activation function. Weights between input to hidden layer and hidden to output layer are represented by matrices $W1$ and $W2$ respectively. The synchronization of both weight matrices between the learning network 35 and adjustment network 16 is triggered by a change of state from low to high in signal RDY. Once this condition occurs, new weights $W1'$ and $W2'$ calculated from learning network 35 will be downloaded into $W1$ and $W2$ respectively.

A denormalizer 17 (Fig. 9) then converts $\Delta\hat{u}(k)$ to unnormalized values $\Delta u(k)$, which functions inversely to normalizer 1 indicated at 15 in Fig. 7. Fig. 9 shows the computations involved in the denormalizer 17. To obtain the new values of the registers, the current register values are offsetted by an amount $\Delta u(k)$. This task is accomplished by the block labelled 18 and Fig. 10 illustrates this simple process. Once all four new register settings $u_b(k+1)$, $u_c(k+1)$, $u_t(k+1)$ and $u_s(k+1)$ are available, the micro-controller 7 will request a "data write" to the I²C bus. If bus-mastering is successful, data in the EEPROM memory on the TV chassis board 10 will be overwritten by the new values $u(k+1)$. Following this, the next adjustment cycle can ensue.

The on-line learning mechanism is instilled into the circuit through the use of the adjustment learner referenced as 40 in Fig. 4. Respective components residing in the adjustment learner 40 will be described next. The set-point database 32 provides a storage of p latest observations organized as $(u(k), y(k))$ pairs. Internally, the database is implemented as a list of depth p . Fig. 11 depicts the list continuously accepting the latest observation $(u(k), y(k))$. If the current time index k is less than p (where k starts from 0), $(u(k), y(k))$ will be inserted into the head of the list. Otherwise, a deletion of the pair $(u(k-p), y(k-p))$ from the tail of the list will follow in addition to the mentioned insertion. Therefore, the set-point database 32 will not experience an increasing amount of data as time progresses.

The STATUS FLAG 53 in this embodiment represents a binary signal from the transport mechanism indicating whether a new TV chassis board 10 is currently in place. When a TV chassis board 10 is first presented to the system, STATUS FLAG 53 is pulled high. This causes the switch in Fig. 11 to be at position R, hence all existing data are discarded as the list in set-point database 32 will be emptied. On the other hand, if the TV chassis board 10 remains unchanged between two successive time steps k , STATUS FLAG 53 is held low. The switch in Fig. 11 will be at position Q, allowing the latest observation pair $(u(k), y(k))$ to be added to the list. The reason for this mechanism is to ensure the integrity of the training set, so that a particular training pair will not originate from two different TV chassis boards 10.

Following the storage of set-points $(u(k), y(k))$, the training pair generator 33 will create a new set of training vectors in the form $(y(k-q)-y(k), u(k-q); u(k-q)-u(k))$ for $q = 1$ to p . Therefore, a total of p training pairs will be generated in every time step k . This process is illustrated in Fig. 12, showing that the resultant training pairs are stored in an array. For the training pairs to be usable by the learning network 35, they have to be normalized first. Normalizer 2 designated by 34 accomplishes this task in an almost identical manner as Normalizer 1 designated 15 in Fig. 7, except that number of inputs have increased. The diagram in Fig. 13 presents the actual calculations involved in normalizer 2 designated 34.

The training pair database 36 of Fig. 4 is implemented as a collection of "bins", which are discrete, disjoint partitions in the input space formed by the vectors $\Delta\hat{y}(k)$ and $\hat{u}(k)$. Since both $\Delta\hat{y}(k)$ and $\hat{u}(k)$ have four dimensions, and are unit-normalized, the input space is a unit-volume hypercube in R^8 space. The hypercube is made up of individual "bins" which house either one or no training pair. The size of each "bin" is determined by the respective boundary values, and hence may not be equal. A training pair will reside in a particular bin if its input vector formed by the concatenation of $\Delta\hat{y}(k)$ and $\hat{u}(k)$ is within the bin's boundary values. At a given time step k , all training pairs in the occupied "bins" will constitute the training set in the training pair database 36. As new training pairs are generated, the "bins" that correspond to the new training pairs will be identified. These bins are then checked to see if they house any existing training pair. If no, the new training pair will be added to the particular bin. Otherwise, the new training pair will overwrite the existing training pair present in the bin. With this method, training pairs in the training pair database 36 can originate from

different TV chassis boards 10. Redundant or similar training pairs are systematically discarded through the overwriting mechanism, while novel training pairs are added.

Once the updating of the training set in training pair database 36 is completed, the training set will be transferred to the learning network 35 for supervised learning through weight optimization. The learning network 35 has identical architecture as the adjustment network 16 shown in Fig. 8. However, actual training is delayed until a minimum number of training pairs is available. In this embodiment, the value is set at 60% of the total number of bins available. The signal CNT from the training pair database 36 keeps track of the number of training pairs available. Once CNT reaches 60% of the total number of bins, training in the learning network 35 is allowed to proceed. This strategy is to prevent the neural network from attempting to control a TV chassis board 10 whose relationship cannot be faithfully represented by a small set of data-points.

Since the adjustment network 16 copies its weight parameters $W1$ and $W2$ from $W1'$ and $W2'$ in learning network 35, some form of synchronization has to be incorporated for a successful transfer. Signal RDY caters for such a purpose. Fig. 14 illustrates how the signal RDY relates to major activities involving the TV chassis board 10 and measurement interface 8, adjustment controller 20 and adjustment learner 40. Fig. 14 also serves the purpose of explaining the concurrent operations involving the components of the invention. The vertical dotted lines in Fig. 14 represent the boundaries of the discrete time steps k . Four time steps are shown from $k=m$ to $k=m+3$.

All activities involved are described with reference to the individual timeslots in Fig. 14. Time-slots without a character inserted therein denote "no operation" in the corresponding component. A complete adjustment and learning cycle commences with the termination of training in the learning network 35, signified by the end of activity in time-slot L2. Following this, RDY will become active (high) for a short period of time before reverting to the inactive state again. During this period of active state, weight parameters $W1'$ and $W2'$ in the learning network 35 are transferred to adjustment network 16. The processes responsible for the function are denoted as C4 in the adjustment controller 20 and L3 in the adjustment learner 40. While the weight parameters are transferred, new values for the four registers $u_b(k)$, $u_c(k)$, $u_r(k)$ and $u_s(k)$ previously computed are written onto the PC bus 3 using the micro-controller 7. Time-

slot P1 is allocated in the measurement interface 8 and TV chassis board 10 for this process. Measurement of the four TV signals $y_b(k)$, $y_c(k)$, $y_t(k)$ and $y_s(k)$, from the TV chassis board 10 are preformed by the TV digitizer 2 of Fig. 4 in time-slot P2. During the time-slot P2, the adjustment controller 20 and adjustment learner 40 are in idle state.

Then, the new register-signal pair measured are transferred to the adjustment controller 20 and adjustment learner 40 during C1 and L1 respectively. Operations in the adjustment controller 20 proceed independently of the adjustment learner 40. In time-slot C2 of the adjustment controller 20, the error signal $e(k)$ is computed and then normalized together with $u(k)$ before being fed to the learning network 16. During C3, the adjustment controller 20 determines the new register values $u_b(k+1)$, $u_c(k+1)$, $u_t(k+1)$ and $u_s(k+1)$ and load the values into the micro-controller 7. While operations in C2 and C3 take place in the adjustment controller 20, the adjustment learner 40 runs the operation specified in time-slot L2. The training sets are updated and the learning network 35 performs Conjugate-Gradient training in L2. The adjustment and leaning cycle will restart again when the training ends.

The criteria for termination of adjustment of a particular TV chassis board 10 is specified by the TV manufacturer. Once the absolute value of the error signal $e(k)$ drops below a specified tolerance, the TV chassis board 10 is deemed to have been tuned successfully. Fig. 15 shows a flowchart summarizing the steps performed by the adjustment controller 20 during a typical adjustment cycle for TV chassis boards 10.

When a new TV chassis board 10 is presented, the time index k is reset to 0 as shown in step 101. In step 102, the signal RDY is monitored to determine whether weights represented by $W1$ and $W2$ are to be updated by $W1'$ and $W2'$ from learning network 35. Measurement of the latest register values $u(k)$ from the PC bus 3 is accomplished in step 103. Next, the TV signals $y(k)$ are captured from the TV chassis board 10 in step 104. Error signal $e(k)$ is calculated from the difference between $y(k)$ and y° in step 105. When $e(k)$ is available, its magnitude is tested against a user specified tolerance in step 106. If magnitude of $e(k)$ falls below the tolerance set, tuning is completed and step 107 is executed. In step 107, STATUS FLAG 53 is pulled high to show that the next in-coming TV chassis board 10 is a new one. Otherwise, the TV chassis board 10 remains in place and both the $\hat{e}(k)$ and $\hat{u}(k)$ are normalized in step 108. In step 109, the corrections to the register values $\Delta u(k)$ are determined by propagating $u(k)$ and $e(k)$ through the adjustment network 16 and denormalizer 17. To arrive at the

new register values $u(k+1)$, the current register values $u(k)$ are offsetted by an amount $\Delta u(k)$ in step 110. The new register values $u(k+1)$ are written to the I²C bus 3 using the micro-controller 7 in step 111. Finally, step 112 increments the discrete time index k before the process starts at step 102 again.

In Fig. 16, a flowchart is presented to illustrate the operations pertaining to the adjustment learner 40 during one learning cycle for TV chassis boards 10. When the cycle starts, step 115 checks the STATUS FLAG 53 to determine whether a new TV chassis board 10 is in place. If affirmative, the time index k is initialized to 0 in step 116 and the set-point database 32 is then emptied in step 117. On the other hand, if the STATUS FLAG 53 signifies that TV chassis board 10 remains unchanged from the one in previous time step $k-1$, steps 116 and 117 are bypassed in the algorithm. In step 118, the latest register values $u(k)$ on the I²C bus 3 are captured using the microcontroller 7. Next, the TV signals $y(k)$ are measured from the TV chassis board 10 in step 119 using the TV signal digitizer 2. In step 120, the newly acquired observation pair $(u(k), y(k))$ is added to the set-point database 32. By performing vector subtraction, new training pairs in the format $(y(k), u(k); u(k))$ are created by the training pair generator 33 in step 121. All newly created training pairs are normalized by the normalizer 2 circuit 34 in step 122. The corresponding "bins" for the new training pairs are then identified and their contents updated accordingly in step 123. Since the training pair database 36 encompasses training pairs in all "bins", its contents are also refreshed. At the same time, CNT counter is incremented to reflect any addition to the training pair database 36. In step 124, CNT is checked to see if it is greater than 60% of the total bin count. If not, steps 125 to 128 are bypassed and k is increased by 1 in step 129 before restarting at step 115. Otherwise, the learning network 35 is allowed to undergo Conjugate-Gradient weight optimizations based on the updated training set in step 125. When training terminates in the learning network 35, the signal RDY is set to active state (high). This is done in step 126. The purpose of RDY is to inform the adjustment network 16 within the adjustment controller 20 that weights $W1'$ and $W2'$ in the learning network 35 is ready for transfer to weights $W1$ and $W2$ in adjustment network 16. The weight transfer is carried out in step 127, and RDY can revert to inactive state (low) subsequently in step 128. The learning cycle is completed after incrementing k by 1 in step 129. The next cycle then starts at step 115 again.



CLAIMS

1. An on-line adaptive controller for sequentially tuning a plurality of electronic circuit boards, said controller comprising:

(a) A reference signal generator for generating a desired response for at least one variable to be controlled;

(b) an error signal generator connected to said reference signal generator and the circuit board to be tuned and arranged to output an error signal indicative of the difference between said desired response and the actual value of said variable(s) to be controlled;

(c) an adjustment calculator including a neural network, said calculator being connected to said error signal generator, and arranged to output a weighted control signal to said circuit board in response to the error signal and the current value of the control signal;

(d) a training pair manager connected to said circuit board and said adjustment calculator to receive said weighted control signal and the actual value of said variable(s) to be controlled, and outputting to a neural network based learning network a corresponding updated training set including new and aggregated training pairs;

(e) said neural network based learning network being independent of, but having substantially identical architecture to, the neural network of said adjustment calculator and being connected to said adjustment calculator to determine the transfer function of its neural network by updating its internal weight parameters; and

(f) synchronising means interconnecting said learning network and said adjustment calculator to synchronise said weight parameter updating.

2. The controller as claimed in claim 1, wherein the training pair manager includes means for storing new observation data from different electronic circuit boards to be tuned, means for generating new training pairs by computing the vector difference between the latest observation and those previously stored, and means for updating the training set by overwriting existing training pairs with new training pairs or appending new training pairs.

3. The controller as claimed in claim 1 wherein said training pair manager receives as inputs the current value of said control signal, the difference between said variable(s) to be controlled measured at two different discrete time steps, and a teaching signal

This Sheet Has Been Replaced



THIS SHEET HAS BEEN
REPLACED



comprising the difference between said control signals measured at two different discrete time steps.

4. The controller as claimed in any one of claims 1-3, wherein said electronic circuit board is a TV circuit chassis board and the variable(s) to be controlled comprise one or more of brightness setting, contrast setting, tint setting and color setting.

5. The controller as claimed in claim 4, wherein each said variable comprises the setting of a corresponding register of an EEPROM memory in said TV circuit chassis board.

6. The controller as claimed in any one of the preceding claims including means to preserve said training set between successive circuit boards.

7. A method of sequentially tuning a plurality of electronic circuit boards, said method comprising the steps of:

- (a) calculating an adjustable weighted control signal based upon a generated error between a reference signal and an actual value of at least one variable to be controlled;
- (b) using a first neural network in the calculation of said control circuit;
- (c) using an independent, but substantially identical neural network based learning network to vary the weighting of said control signal; and;
- (d) training said learning network with the weighted control signal and the actual value of said variable(s) to be controlled to adaptively vary said control signal weighting.

8. A method as claimed in claim 7 wherein said circuit boards are TV circuit chassis boards, said method comprising the steps of:

- (a) measuring the readings for controlled variables from TV chassis boards using a TV digitizer circuit;
- (b) reading the values for the control parameters from TV chassis boards using a micro-controller interfaced by the Inter IC (1^2C) bus;
- (c) converting the controlled variables to error signals;
- (d) normalizing the current values of control parameters and error signals, deriving the corrections to the control parameters using neural network based controller and denormalizing the resultant corrections;





(e) converting the corrections to the control parameters to new settings for the control parameters;

(f) writing the new settings for the control parameters onto the TV chassis boards using a micro-controller interfaced by the Inter IC (I²C) bus.

9. The method as claimed in claim 8 and comprising the further steps of:

(g) deciding to either reset or update a dynamic list based upon a monitored status;

(h) organising the control variables and read variable values as observation pairs and storing them in said dynamic list;

(i) creating new training pairs by computing the vector difference between observation pairs in the said dynamic list and the most recent observation pairs;

(j) normalizing all training pairs duly created in step (i);

(k) updating the relevant partitions occurring in the input space which correspond to the newly created training pairs;

(l) updating a count;

(m) determining whether the count is greater than a pre-defined percentage of the total number of partitions allowed in the input space;

(n) optimizing the performance of said neural learning network using Conjugate-Gradient learning; and

(o) altering the weighting of the adjustably weighted control signal by over-writing current first neural network values with new values obtained in step (n).

THIS SHEET HAS BEEN
REPLACED

This Sheet Has Been Replaced

