



166166

Methods and devices for compensating for periodic disturbances
in servo-systems

The present invention relates to a method for controlling a system, particularly a servo system, such as the read/write head of a disk drive. The invention
5 further relates to a controller which operates the method, and a servo-system incorporating the controller.

Periodic disturbances appear in many servo systems and degrade system performance. A good example is the servo system of a read/write head in a disk drive for data storage, where the main function of the servo is to maintain
10 the head accurately on a selected track of the spinning disk. The positioning inaccuracy due to the influence of the periodic disturbance is referred as Repeatable Run-Out (RRO), which forms one of the main obstacles to improving the system positioning accuracy. Figure 1 shows the position error signal of a 3.5" hard disk drive after the read-write head comes to the desired
15 track and tries to maintain its position at that track, a processes which is referred to as track following. Due mainly to eccentricity of the spindle disks there is a periodic disturbance in the position of the head, and consequently a conventional servo can ensure the residual error only to within $\pm 25\%$ of the track width, whereas the required position accuracy should be less than $\pm 10\%$
20 of the track width. In disk drives in which the disks are removeable, the RRO is much larger resulting from the larger eccentricity of the disks. This kind of problem with periodic disturbances also occurs in rotational machinery, including machine tools, and robot manipulators doing a repetitive task. Furthermore, some nonlinearities with actuators can be modeled as periodic
25 disturbances.

A common property of these disturbances is that they are periodic (i.e. repetitive) and the period is either known or can discovered by analysis or identification procedures. It can be seen that the residual error $e(t)$ as shown



G00002

in Figure 1 is periodic, and the fundamental frequency of the residual error is 75 Hz, which corresponds exactly with the spindle speed of the disks.

A conventional way of cancelling RRO is to use notch filters as part of a feedback loop. Since the notch filter is inserted in series in the feedback loop, it has effect on all signals at the notch frequency in the loop. Therefore, insertion of the notch filter into the closed loop changes the system characteristics, which makes the design of the servo controller more difficult than before. Moreover, the main drawback of using notch filters is that the method is model-based: the notch filters are designed with fixed parameters selected based on the knowledge about the disturbances. Since the disturbances may vary with the operational situation, compensation by notch filters is not always effective.

Another proposed solution to RRO is the Adaptive Feedforward Cancellation (AFC) [1]-[3]. Unlike the notch filter method, the AFC works in a feedforward path, which does not change the system characteristics. However, it has been found that an AFC algorithm designed to reject a certain number of frequency components may generate higher-order harmonics [3], which may degrade the compensation performance. Furthermore, a phase shift is required in the "modified AFC" while an inaccurate identification of the phase shift leads to large residual error [1], [2]. In addition, the realization of the AFC needs the *sin* and *cos* functions, which are usually implemented by a look-up table in a digital signal processor (DSP). The look-up table occupies large memory space and makes the programming tedious.

In [4] the present inventor proposed an Observer-based Feedforward Compensation (OFC) method. The RRO is treated as additional system states to be observed by a state observer. The observed signals are then used to develop a feedforward compensation loop to remove the RRO influence. A drawback of OFC is that it needs two extra state variables for each harmonic

component of the RRO; this makes it difficult to implement compensation for enough harmonic components.

Another scheme to reject periodic disturbances is the repetitive controller, which is compared with the AFC in [5]. As stated there, a repetitive controller
5 can compensate for higher harmonics than AFC and the computation burden of a repetitive controller is only a quarter of that of an AFC. However, the compensation effect by a repetitive controller for each harmonic is smaller than that by an AFC. The repetitive controllers used in this citation gave a
10 reduction ratio of 45 dB for the fundamental harmonic and an even lower reduction for the other higher harmonics, whereas the AFC has a large reduction ratio of more than 60 dB for each compensated harmonics.

Furthermore, all the above methods are implemented by software in a DSP or a microprocessor. Therefore, the limitations of the DSP or the microprocessor, (e.g. limited computation speed and/or word length) may limit their ability to
15 compensate for disturbances.

The present invention seeks to provide new and useful methods and devices for controlling a servo-system, particularly ones which can compensate for the periodic disturbances in servo systems effectively, robustly, simply and economically.

20 In general terms, the present invention proposes that a servo-system subject to periodic noise is controlled using a neural network which includes two layers: a hidden layer and an output layer. Pairs of neurons in the hidden layer are connected by self-feedback such that each pair of neurons oscillates at a respective harmonic of a basic frequency ω . The outputs of the neurons
25 of each pair are respectively the sine and cosine of the corresponding harmonic. The output layer forms a weighted sum of the output of the hidden layer neurons, which in turn is used to control the servo-system. The neural network is tuned such that the output of the output layer matches the periodic

disturbances in the servo-system, and thus can be used to compensate for them. If the highest harmonic order of the periodical disturbances to be compensated is n , the neural network requires $(2n+1)$ neurons in the hidden layer and a single neuron in the output layer.

- 5 The output of the neural network is preferably combined with that of a servo-controller (e.g. of a conventional design), to form a combined control signal which is input to the system being controlled.

The invention is based on the inherent property of periodic disturbances that they are expandable into trigonometric series. In effect, each pair of neurons
10 in the hidden layer may compensate for one of the components of the trigonometric series. Thus, with a sufficiently large number of hidden layer units, periodic disturbances of any given order can be accurately compensated for.

The neural network may be tuned automatically, that is it has learning
15 capability to learn and adapt the environment variations, such as changes in the periodic noise.

As described below, the method and control device may be implemented not only by software in a DSP but also by hardware so that the limitations of the servo DSP do not limit the compensation available.

- 20 In particular, the control method and device can be implemented by analog circuitry, preferably integrated circuitry such as VLSI. In particular, the neural network and its learning system can be integrated in a chip, such as an additional part integrated in an existing driver chip.

An advantage of the present invention over the AFC is that the sin/cos look-up
25 tables in the AFC are replaced by the self-feedback in the hidden layer. This may make the invention more simple and reliable.

Furthermore, due to its learning mechanism, the neural network can deal with changing operation situations. This makes preferred embodiments of the invention more robust.

Furthermore, because of the clear physical meaning in structure, the present invention makes it possible to compensate for any order of harmonics of the periodic disturbances by simply adding more neurons. This renders the preferred embodiments of the invention more flexible.

An non-limiting embodiment of the present invention will now be described in detail for the sake of example, with reference to the following figures in which:

10 Fig. 1 shows the influence of a periodic disturbance in a servo system of a hard disk drive;

Fig. 2, which is composed of Figs. 2(a) and 2(b), is (a) a schematic diagram of the structure of an embodiment of this invention controlling the servo system of a hard disk drive, and (b) an equivalent diagram representing the periodical disturbance as a force signal;

15

Figure 3 shows the structure of the neural network of Fig. 2 schematically;

Fig. 4 shows schematically the neural network of Fig. 3 and the learning mechanism, in that case that the neural network is represented by a continuous system;

20 Fig. 5 is schematic block diagram of the neural network of Fig. 3, and its the learning mechanism, implemented in DSP software; and

Figure 6 shows a possible implementation of the neural network of Fig. 3 by analog VLSI.

Referring firstly to Figure 2 (a), an embodiment of the invention is shown. This may for example be the servo system of a hard disk drive.

It comprises a system 210 being controlled, corresponding to the read/write head. The system is subject to disturbance r . Note that this signal, and all the
5 others discussed below, are functions of time. A measurement signal y_m is output from the system 210, and indicates the state of the system 210. It is compared with a reference signal y^* , to give a position error signal e which is the input of a nominal servo controller 200, which provides a command signal u to drive an actuator. The nominal servo controller is a servo controller
10 designed for the nominal system, i.e. system 210 without the disturbance r .

The system further includes a unit 220 comprising a neural network 222. In this text the neural network, which is described in detail below, is referred to as a hybrid trigonometric neural network (HTNN). It has a hidden layer made up of $(2n+1)$ neurons, which have respective outputs given by a vector $\mathbf{X}$, and
15 an output layer which produces a single output f_N . The HTNN 222 is controlled on the basis of a set of $(2n+1)$ weights which are written as a vector $\mathbf{W}$. The vector $\mathbf{W}$ is set by a learning mechanism (LM) 221, which receives the outputs $\mathbf{X}$ of the hidden layer neurons, as well as the instantaneous value e .

The output u of the nominal servo controller 200 is added to the output f_N of
20 the HTNN 222, to form a control signal which is the input to the controlled system 210. In this analysis, the controlled system 210 is modelled as two integrators 211 and 212.

In the case that the system 210 is a hard disk drive (HDD), it comprises a moveable actuator, arm with suspension, a driving means to provide
25 displacement for positioning the actuator, and a voice coil motor in the HDD. None of these components are shown here, but all may be of a conventional construction and will be clear to a skilled person.

The periodic disturbance called Repeatable Run-Out (RRO) here is denoted by r , which in the case of a HDD appears as a positional disturbance caused by eccentricity of the disk, deformation of the spindle axis, or imperfection of the servo writing process.

- 5 Since the RRO is periodic, it can be approximated by a trigonometric polynomial as

$$r(t) = \sum_{i=0}^n R_i \sin(i\omega t + \varphi_i) \quad (1)$$

- Although the RRO appears as a positional disturbance, it is equivalent to a force disturbance $f(t)$ as shown in Figure 2(b) with the Laplacian
10 transformation $F(s)$.

$$F(s) = \frac{s^2}{k_v k_y} R(s) \quad (2)$$

where $R(s)$ is the Laplacian transformation of the RRO $r(t)$. Therefore,

$$\begin{aligned} f(t) &= -\sum_{i=1}^n \frac{i^2 \omega^2 R_i}{k_v k_y} \sin(i\omega t + \varphi_i) \\ &= \sum_{i=1}^n F_i \sin(i\omega t + \varphi_i) \end{aligned} \quad (3)$$

15

where $F_i = -i^2 \omega^2 R_i / k_v k_y$. The first term with the index of zero in (1) disappears after the double derivations since it is a constant. Equation (3) can be further written as

$$\begin{aligned}
f(t) &= \sum_{i=1}^n F_i (\sin i\omega t \cos \varphi_i + \cos i\omega t \sin \varphi_i) \\
&= \sum_{i=1}^n ((F_i \cos \varphi_i) \sin i\omega t + (F_i \sin \varphi_i) \cos i\omega t) \\
&= \sum_{i=1}^{2n} w_i x_i(t) \\
&= \mathbf{W}^T \mathbf{X}(t)
\end{aligned} \tag{4}$$

where $\mathbf{X}(t) \in \mathbb{R}^{2n}$, $\mathbf{W} \in \mathbb{R}^{2n}$, and they are given by:

$$\begin{aligned}
\mathbf{X}(t) &= [\sin \omega t \quad \sin 2\omega t \quad \cdots \quad \sin n\omega t \quad \cos \omega t \quad \cos 2\omega t \quad \cdots \quad \cos n\omega t]^T \\
\mathbf{W} &= [F_1 \cos \varphi_1 \quad F_2 \cos \varphi_2 \quad \cdots \quad F_n \cos \varphi_n \quad F_1 \sin \varphi_1 \quad F_2 \sin \varphi_2 \quad \cdots \quad F_n \sin \varphi_n]^T
\end{aligned}$$

By the property of the trigonometric function, we have the following matrix
5 dynamic equation

$$\dot{\mathbf{X}}(t) = \mathbf{A}\mathbf{X}(t) \tag{5}$$

where

$$\mathbf{A} = \begin{bmatrix} 0 & A \\ -A & 0 \end{bmatrix}, \quad A = \begin{bmatrix} \omega & 0 & \cdots & 0 \\ 0 & 2\omega & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & n\omega \end{bmatrix}, \tag{6}$$

10 and $\mathbf{A} \in \mathbb{R}^{2n \times 2n}$, $A \in \mathbb{R}^{n \times n}$.

Equations (4) and (5) show that the periodic disturbance can be expressed by a trigonometric polynomial, which is graphically represented by a neural network in Figure 3. The network has two layers: a hidden layer with $(2n+1)$ neurons and an output layer made up of just one neuron. The outputs of the
15 neurons except neuron 303 of the hidden layer labelled by $x_1, \dots, x_n, x_{n+1}, \dots, x_{2n}$ are equal to respective components of $\mathbf{X}$ in Eqn. (4).

In Figure 3, each neuron of the hidden layer except neuron 303 performs the simple function of multiplying its input by a respective value $a_1, \dots, a_n, a_{n+1}, \dots, a_{2n}$ (this multiplication is shown as element 300) and applying the result to an integrator 301, to produce the output of the hidden layer neuron. Neuron 303 is a special neuron which gives a constant output of 1 and works as a bias for the output layer. Neuron 303 can be treated equally to the other neurons in the hidden layer but with its $a_0=0$, thus we have $\mathbf{X} \in \mathbb{R}^{2n+1}$ and $\mathbf{A} \in \mathbb{R}^{(2n+1) \times (2n+1)}$.

Each output $x_0, \dots, x_n, x_{n+1}, \dots, x_{2n}$ from the hidden layer is applied to the output layer which consists of a single neuron which multiplies (in multipliers shown as 304) each input by a respective adjustable weight $w_0, \dots, w_n, w_{n+1}, \dots, w_{2n}$, to give respective results 306, and sums the results 306 (using addition element 307) to get the output 308 which is $f_N(t, \mathbf{W}(t))$.

The values of the adjustable weights $w_0, \dots, w_n, w_{n+1}, \dots, w_{2n}$ can be varied (shown in Fig. 3 as operation 305) by the learning mechanism described below.

This network is very similar to a so-called "radial neural network" in structure, since it has only one adjustable weight in each feedforward path, and no exit function before output (i.e. the output of the output neuron is just the weighted sum of the inputs, not some function of that weighted sum). The difference between the present network and a radial neural network is that the network in Figure 3 obtains its input to the output layer by its own dynamics and in the form of trigonometric functions. It is for this reason that we refer to the network in Figure 3 as a Hybrid Trigonometric Neural Network (HTNN).

On the HTNN, we have two remarks:

1. The HTNN can approximate any periodic signals. This statement can be mathematically expressed as: for any continuous periodic $f(t+T) = f(t)$ with

a period T and for any $\delta > 0$, then there exists a finite value of n and a weight vector $\mathbf{W}^*$ such that

$$|f_N(t, \mathbf{W}^*) - f(t)| < \delta. \quad (7)$$

2. Among all trigonometric polynomials with period T and the order less than n , the HTNN gives the best possible mean square approximation to f .

We will now discuss the learning mechanism of the HTNN of Fig. 3, and system stability.

From Figure 2 (b), if $y = y_m$, the dynamics of the HDD servo without the HTNN can be described as

$$\ddot{y}(t) = k_v k_y u(t) + k_v k_y f(t). \quad (8)$$

If the nominal servo-controller 200 is designated as a linear PD controller, we have

$$u(t) = (k_1 \dot{e}(t) + k_0 e(t)) / k_v k_y, \quad (9)$$

where $e = y^* - y$ is the position error.

- 15 To write the gains of the controller as

$$\begin{cases} k_0 = k_c \beta_0 / \beta_1 \\ k_1 = k_c + \beta_0 / \beta_1 \end{cases} \quad (10)$$

where $k_c > 0$, $\beta_0 > 0$ and $\beta_1 > 0$ are factors for a composite error defined as

$$\varepsilon(t) = \beta_0 e(t) + \beta_1 \dot{e}(t). \quad (11)$$

Thus, by differentiating (11) with the consideration of (8) to (10) we have

$$\dot{\varepsilon}(t) = -k_c \varepsilon(t) + \beta_1 f_R(t). \quad (12)$$

5 where $f_R(t) = -k_v k_y f(t)$.

Taking the output $f_N(t, W(t))$ of the HTNN 222 into account, we have

$$\dot{\varepsilon}(t) = -k_c \varepsilon(t) + \beta_1 (f_R(t) - f_N(t)). \quad (13)$$

For the stability design, a Lyapunov function is constructed as

$$10 \quad V(t) = \frac{1}{2\beta_1} \varepsilon^2(t) + \frac{1}{2\eta} E_W^T(t) E_W(t) \quad (14)$$

where $E_W(t) = W^* - W(t)$ is the weight error.

In the case that the HTNN is operated in continuous time (e.g. by analogue circuitry), the learning rule for the HTNN is here chosen to be:

$$15 \quad \dot{W}(t) = \eta \varepsilon(t) X(t), \quad (15)$$

where $W \in \mathbb{R}^{2n+1}$ and $X \in \mathbb{R}^{2n+1}$.

Thus, the operation of the HTNN and the learning algorithm is as shown schematically in Figure 4, where box 400 represents the HTNN and box 401 shows the operation of the learning mechanism. The HTNN has a continuous

loop of multiplying $\mathbf{X}$ by $\mathbf{A}$, and $\mathbf{X}$ is also fed back to the learning mechanism 401. The time derivative of $V(t)$ is thus

$$\begin{aligned}
 \dot{V}(t) &= \frac{1}{\beta_1} \varepsilon(t) \dot{\varepsilon}(t) - \frac{1}{\eta} \mathbf{E}_{\mathbf{W}}^T(t) \dot{\mathbf{W}}(t) \\
 &= -\frac{k_c}{\beta_1} \varepsilon^2(t) + \varepsilon(t) (f_R(t) - f_N(t)) - (\mathbf{W}^{*T} - \mathbf{W}^T(t)) \varepsilon(t) \mathbf{X}(t) \\
 &= -\frac{k_c}{\beta_1} \varepsilon^2(t) + \varepsilon(t) (f_R(t) - f_N(t) - f_N^*(t) + f_N(t)) \\
 &\leq -\frac{k_c}{\beta_1} \varepsilon^2(t) + |\varepsilon(t)| |f_R(t) - f_N^*(t)| \\
 &\leq -\frac{k_c}{\beta_1} \varepsilon^2(t) + |\varepsilon(t)| \delta
 \end{aligned} \tag{16}$$

5 Thus, we have $\dot{V}(t) \leq 0$, when

$$|\varepsilon(t)| \geq \frac{\beta_1}{k_c} \delta \tag{17}$$

Therefore, the system composite error converges towards $\beta_1 \delta / k_c$, which in turn means the error $e(t)$ converges by the linear differential equation:

10

$$\dot{e}(t) + \frac{\beta_0}{\beta_1} e(t) \geq \frac{\delta}{k_c}. \tag{18}$$

We have the solution of (18)

$$e(t) \geq \left(e(0) - \frac{\beta_1 \delta}{\beta_0 k_c} \right) \exp\left(-\frac{\beta_0}{\beta_1} t \right) + \frac{\beta_1 \delta}{\beta_0 k_c}. \tag{19}$$

Eqn. (19) shows that the system error $e(t)$ converges exponentially with an exponential constant of β_0 / β_1 from its initial value $e(0)$ with a bias of $\beta_1 \delta / (\beta_0 k_c)$ towards a positive constant $\beta_1 \delta / (\beta_0 k_c)$.

We now turn to the question of how to select the parameters of the HTNN and the learning mechanism.

The initial state of the hidden layer neurons of the HTNN, that is $X(0)$, should be assigned to ensure the state $X(t)$ remains trigonometric, i.e.

$$\begin{cases} X_i(0) = 0, & i = 1, n, \\ X_i(0) = 1, & i = n+1, \dots, 2n. \end{cases} \quad (20)$$

The initial values of the weights $W(0)$ in HTNN can be set to be zero to make the HTNN have a zero initial output,

$$W(0) = 0. \quad (21)$$

The HTNN order n affects the approximation accuracy δ and further affects the whole system compensation performance. Generally, a large n results in a small δ and thus a good compensation. However, the HTNN order n is limited in practice. For example, it may be limited by the scale of the integration when the HTNN is implemented by analog VLSI, or limited by the computation capacity of the DSP when the HTNN is implemented by software. A practical way to select the HTNN order n is by a harmonic analysis to the error signal. From the contents of the harmonic components of the error signal, a suitable n can be chosen to ensure the HTNN can cancel all the significant harmonic components. Another way to set n is by increasing n from a initial value until an acceptable compensation performance is achieved with consideration of

the hardware or software limitations. This process can be carried out before the HTNN is set to work or by an on-line adaptation.

We now turn to how to set the parameters β_0 , β_1 , and k_c .

From (10) we have

$$5 \quad \begin{cases} k_c^2 - k_1 k_c + k_0 = 0 \\ \beta_0 / \beta_1 = k_1 - k_c \end{cases} \quad (22)$$

Assuming the linear controller is designed to make the linear system work in critical state, i.e.

$$k_1^2 = 4k_0, \quad (23)$$

10 we have

$$\begin{cases} k_c = k_1 / 2, \\ \beta_0 / \beta_1 = k_1 / 2. \end{cases} \quad (24)$$

If we set $\beta_1 = k_1$, then $\beta_0 = 2k_0$. Thus, from (9) and (11), we have

15

$$\varepsilon(t) = u(t) + k_0 e(t). \quad (25)$$

Using (25) in the learning mechanism avoids the measurement of the derivative of $e(t)$, which is usually problematic in practice.

We now turn to the issue of the implementation of the embodiment in either hardware or software.

Firstly, an implementation in software using a DSP and by hardware by analog VLSI is described in detail in the following. In this case, the continuous equations discussed above are no longer appropriate, and instead we transform Eqn. (5) as follows

$$\begin{cases} \mathbf{X}_d(k+1) = \mathbf{D}\mathbf{X}_d(k) \\ \mathbf{D} = e^{AT} \end{cases} \quad (26)$$

where T is the sampling time and $\mathbf{X}_d(k) \in \mathbb{R}^{2n+1}$, $\mathbf{D} \in \mathbb{R}^{2n \times 2n}$. The initial value of $\mathbf{X}_d(k)$ should be

$$\mathbf{X}_d(0) = [\mathbf{0}_{1 \times n} \quad \mathbf{I}_{1 \times n}]^T$$

The learning rule (15) is replaced with:

$$\Delta W(t) = \eta \varepsilon(t) X(t), \quad (27)$$

The schematic block diagram of the HTNN with its learning mechanism realized by DSP software is shown in Figure 5, where 500 is the network body and 501 is the learning mechanism. In this case, the overall structure resembles that of Fig. 4, but in the block 500 matrix $\mathbf{A}$ is replaced by matrix $\mathbf{D}$ and the integrator is replaced by a delay unit (labelled " z^{-1} "), and similarly, in the block 501 corresponding to the learning mechanism, the integrator is

replaced by a delay unit z^{-1} , and the output of the block 501 is the sum of the delay unit (i.e. W at a previous cycle, with the result of equation (27)).

Secondly, we will discuss an implementation of the embodiment by analog VLSI. In this case the equations (5) and (15) remain valid and are not
 5 replaced by (26) and (27). An example implementation is shown in Figure 6, following the design methodology introduced in the publications [6], [7].

The top left of Figure 6 introduces a component 600 called a transconductance amplifier (discussed in [7]). It has an output current i_o which is a function of the input voltage v_i between two of its input terminals, a bias
 10 current i_b to another terminal, and a gain K , given by:

$$i_o = Kv_i + i_b. \quad (28)$$

Fig. 6 also introduces a component 601 called a four-quadrant multiplier having an output current i_o which is a function of the two input voltages v_{i1} and
 15 v_{i2} and a bias current i_b given by:

$$i_o = v_{i1}v_{i2} + i_b, \quad (29)$$

The upper part of Fig. 6 is the network body and the lower part is the learning mechanism.

20 The network body contains $2n$ neurons marked as 610 in the hidden layer and one neuron 620 in the output layer. The output voltage from the part 610 are part of the inputs of part 620. Part 610 has components 611, 612, 613 and 614 which implement the hidden layer neuron. Part 620 has components 621 and 622 which perform the multiplication of the output of the hidden layer
 25 neuron by a respective weight, and takes the sum to give a voltage output v_o .

A constant voltage v_{x0} marked as 640 through weight w_0 , which is generated by a part corresponding to 630, gives a bias to the neuron 620 by the bias current of any multiplier in 620, e.g. 621 in Figure 6. Part 630 generates the weight for use by multiplier 621.

- 5 The error signal $e(t)$ is generated using components shown as amplifier 650, which is connected in the resistive arrangement shown to the two inputs y and y^* .

We will now analyse this behaviour carefully. Using the definition of the transconductance amplifier 612 and taking into account the resistor 613 with a
10 value R , we obtain:

$$dv_{x1} = KR a_1 v_{x1}. \quad (30)$$

where the feedback matrix $[a_i]$ is determined by connecting or disconnecting each node in component 610. Using the follower-integrator 614, which
15 consists of a transconductance amplifier and a capacitor C , we obtain

$$\dot{v}_{x1} = \frac{C}{K} dv_{x1} = CR a_1 v_{x1}. \quad (31)$$

for a correct selection of K and R .

The multiplier 621 multiplies the output of the hidden layer neuron by multiplying it with the learned weight w_1 according to the formula:

20

$$i_{o1} = v_{x1} w_1 + i_{b1},$$

$$= v_{x1} w_1 + w_0 \quad (32)$$

where the weight is obtained by the learning mechanism 630 and is $i_{b1} = w_0$.

The whole output of the HTNN is obtained by taking the sum of the current contributions from all $2n$, and transforming the current i_o into a voltage signal v_o using amplifier 622.

$$v_o = Ri_o = R \sum_{k=1}^{2n} w_k v_{xk} + i_{bk} = R \sum_{k=1}^{2n} w_k v_{xk} + w_0$$

Each learning component 630 updates the corresponding weight value continuously according to the expression:

$$\dot{w}_1 = \eta \frac{CR}{K} v_{x1} (y^* - y). \quad (33)$$

Thus, the HTNN in equation (4) is realized by analogy VLSI.

As discussed above, the embodiments of the present invention may provide a method and a system which can compensate for the influences of periodic disturbances on the system performance of a positioning servo system. Such a method and such a system have the following advantages:

- They can be realized in a systematic way so that all the required components of the periodic disturbance can easily be compensated.
- They have learning capability to learn and adapt the environment variations.
- They can be implemented not only by software in a DSP but also by hardware so that the limitations of a servo DSP do not affect the compensation effect of the invention.

- They can be implemented by analog VLSI so that the whole HTNN and its learning can be integrated in a chip.

Although only one embodiment of the present invention has been described in detail above, many variations are possible within the scope of the invention as will be clear to a skilled person.

References

The disclosure of the following references is incorporated herein by reference in their entirety:

- [1] A. Sacks, M. Bodson, and W. Messer, "Advanced method for repeatable runout compensation", IEEE Trans. on Magnetics, vol. 31, no. 2, pp. 1031-1036.
- [2] Y. Onuki, H. Ishioka and H. Yada, "Repeatable runout compensation for disk drives using multi-loop adaptive feedforward compensation", SICE '98, Chiba Japan, pp. 1093-1098.
- [3] M. Bodson, A. Sacks and P. Khosla, "Harmonic Generation in adaptive feedforward cancellation schemes", IEEE Trans. on Automatic Control, vol. 39, no. 9, pp. 1939-1994.
- [4] Y. Lou, "Observer-based compensation of repeatable run-outs in hard disk drives", Proceedings of the 9th Sony Research Forum (1999) p453-458
- [5] Y. Onuki and H. Ishioka, "Compensation for repeatable tracking errors in hard drives using discrete-time repetitive controllers", Proceedings of the 6th

International Workshop on advanced motion control, March 2000, Nagoya, Japan, pp 490-495.

- [6] S. P. DeWeerth et al, "A neuron-based pulse servo for motion control",
5 Proc. IEEE Int Conf. Robotics and Automation (Cincinnati, OH), 1990,
pp1698-1703,

[7] C.A. Mead, Analog VLSI and neural systems, Reading MA, Addison-Wesley, 1989.

Claims

1. A method of controlling the behaviour of a system, the method comprising:

5 receiving a measurement signal indicative of the behaviour of a system;

from the measurement signal generating an error signal indicative of an error in the behaviour of the system

10 using the error signal to control a neural network having a hidden layer including a plurality of hidden-layer neuron units, each for outputting a respective predetermined function of its inputs, and an output layer comprising an output layer neuron unit for outputting a weighted sum of the outputs of the hidden-layer neuron units, a plurality of the hidden layer neuron units being arranged in pairs, the output of each hidden layer neuron unit of each pair being input to the other hidden layer neuron unit of the pair;

15 transmitting the error signal also to a servo controller, such as a linear proportional derivative controller, to generate a correction signal, and

combining the correction signal with the output of the output layer neuron unit and using the result as a control signal for controlling the system.

20 2. A method according claim 1 in which, labelling the number of said pairs as n and the pairs as $i=1, \dots, n$, for each i a first one of the pair of hidden layer neuron units outputs a function $i\omega v(h)$ of its input h , and the other one of the i -th pair outputs $-i\omega v(h)$, where v is a predetermined function and ω is a constant.

25 3. A method according to claim 2 in which $v(h)$ is an integral of h over a period of time.

4. A method according to claim 3 in which the hidden layer neuron units are implemented in analog VLSI hardware.
5. A method according to claim 2 in which $v(h)$ is the discrete-time integration of h over a period of time
- 5 6. A method according to claim 5 in which the hidden layer neuron units are implemented in software.
7. A method according to any preceding claim in which said network is controlled by controlling weight values of the weighted sum.
8. A method according to claim 7 in which the weights are updated by an
10 amount which is a proportionality factor multiplied by the instantaneous output of the corresponding hidden layer.
9. A method according to claim 8, in which the proportionality factor is a function of the output of the servo controller and the instantaneous error signal.
- 15 10. A method according to any preceding claim which includes a preliminary step of observing the behaviour of the system to select the number n of pairs of neuron units employed in the hidden layer, and of setting the initial outputs of each of the hidden layer neuron units and the weight values for the selected number of neuron units.
- 20 11. A controller for controlling the behaviour a system, the controller comprising:

an input for receiving a measurement signal indicative of the behaviour of a system;

a comparator for comparing the measurement signal with a reference signal to generate an error signal indicative of an error in the behaviour of the system;

a neural network having a hidden layer including a plurality of hidden-layer neuron units, each for outputting a respective predetermined function of its inputs, and an output layer comprising an output layer neuron unit for outputting a weighted sum of the outputs of the hidden-layer neuron units, a plurality of the hidden layer neuron units being arranged in pairs, the output of each hidden layer neuron unit of each pair being input to the other hidden layer neuron unit of the pair;

a neural network learning mechanism for controlling the neural network based on the error signal;

a servo controller arranged to receive the error signal and to generate a correction signal, and

a controller for combining the correction signal with the output of the output layer neuron unit and using the result as a control signal for controlling the system.

12. A controller according to claim 11 in which, labelling the number of said pairs as n and the pairs as $i=1, \dots, n$, for each i a first one of the pair of hidden layer neuron units is arranged to output a function $i\omega v(h)$ of its input h , and the other one of the i -th pair is arranged to output $-i\omega v(h)$ of its input h , where v is a predetermined function and ω is a constant.

13. A controller according to claim 12 in which $v(h)$ is an integral of h over a period of time.

14. A controller according to claim 13 in which the hidden layer is implemented in analog VLSI hardware.

15. A controller according to claim 12 in which $v(h)$ is the discrete-time integral of h over a period of time.
16. A controller according to claim 15 in which the hidden layer is implemented in digital software.
- 5 17. A controller according to claim 11 in which said neural network learning mechanism is arranged to control the weight values of the weighted sum.
18. A method according to claim 17 in which the neural network learning mechanism is arranged, at each update of the weights, to vary the weights by an amount which is a proportionality factor multiplied by the instantaneous
10 output of the corresponding hidden layer.
19. A controller according to claim 18, in which the proportionality factor is a function of the output of the linear proportional derivative controller and the instantaneous error signal.