Proceedings of the 2003 IEEE/ASME
International Conference on
Advanced Intelligent Mechatronics (AIM 2003)

Phase correction for Learning Feedforward Control

Bas J. de Kruif and Theo J. A. de Vries
Drebbel Institute of Mechatronics,
University of Twente, The Netherlands.
email:b.j.dekruif@utwente.nl

ABSTRACT

Intelligent mechatronics makes it possible to compensate for effects that are difficult to compensate for by construction or by linear control, by including some intelligence into the system. The compensation of state dependent effects, e.g. friction, cogging and mass deviation, can be realised by Learning FeedForward Control. This method identifies these disturbing effects as function of their states and compensates for these, before they introduce an error. Because the effects are learnt as function of their states, this method can be used for non-repetitive motions.

The learning of state dependent effects relies on the update signal that is used. In previous work, the feedback control signal was used as an error measure between the approximation and the true state dependent effect. If the effects introduce a signal that contains frequencies near the bandwidth, the phase shift between this signal and the feedback signal might seriously degenerate the performance of the approximation. The use of phase correction overcomes this problem. This is validated by a set of simulations and experiments that shows the necessity of the phase corrected scheme.

1 INTRODUCTION

The fusion of control engineering and mechanical engineering results in the mechatronic approach to system development, in which the system is designed as a whole, and not in separate blocks. The performance of the controlled system, in terms of the tracking error, can be increased even more by including intelligence into the control. Increasing the performance by including intelligence in the control is pursued by the field of intelligent mechatronics. Effects that are difficult to compensate for by construction or feedback control, e.g. friction and cogging, might be compensated for by using learning control [6].

A well known intelligent control scheme for systems that have to perform repetitive tasks is Iterative Learning Control (ILC) [7, 8]. This control scheme learns a feed-forward signal as a function of the time. Whenever the system starts with its task, which is repetitive, the errors that were measured during the previous execution of the task, are incorporated in the existing feed-forward signal and so fine-tunes the feed-forward signal to compensate for these errors. The kind of errors that can be compensated for are the errors that occur at a predictable moment in time during execution of the task. Commonly, these errors originate from state dependent

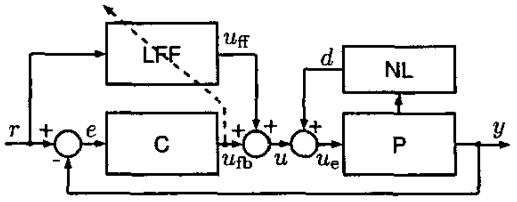

Figure 1: The classic LFFC-scheme

effects, because they will introduce the same error each time the state is passed. The state dependent effects that induce the errors have to be rather smooth if they are to be compensated for by feedforward, because the calculated feedforward signal should compensate for the effect, while this same feedforward signal shall slightly alter the state the plant will pass in the next run. If the magnitude of the effect changes considerably due to the state change induced by the feedforward signal, the effect will not be nullified by the calculated feedforward signal.

A drawback of the ILC-scheme is that it can only be applied if the task is repetitive. If, instead of learning a feedforward signal as a function of time, the state dependent effects are identified as function of their states, the influence these effects have on the behaviour of the plant can be compensated for, independent of the commanded motion. The learning of the effects as function of the states is known as feedback error learning [5] and the control configuration is called Learning FeedForward Control (LFFC) [9, 12].

The above mentioned references on LFFC use the output of the feedback controller as a measure of the error between the true state dependent effects and the approximation thereof. This might be sensible if the influence of these effects contains only frequencies well within the bandwidth of the controlled system, but if this influence contains high frequencies, the control signal is not a good representation of the error between the approximated and the true state dependent effects. The use of this incorrect representee of the approximation error might jeopardise the performance and the stability, as discussed in [12].

In this paper, we will study how the approximation of the state dependent effects can be adapted such that the high frequency components will not lead to incorrect adaptation. This is achieved by using ILC techniques in the calculation of a correct error signal.

To address this problem, first the background on LFFC and ILC will be treated in section 2. With the theory used in ILC it is possible to correct the phase for the learning signals in LFFC and this will be treated in section 3. The method will be validated with a set of simulations and experiments in section 4. A conclusion will be drawn in section 5.

2 BACKGROUND

Because ILC is a common technique, it will be treated concisely. For a thorough treatment of ILC, see e.g. [7, 8]. The part on LFFC will be treated more elaborately. It will be explained what the basic idea is, with emphasis on the conditions under which it will fail. For the interested reader, we refer to [9, 12] and the references therein.

2.1 Learning Feedforward Control

Learning Feedforward Control tries to identify state dependent effects, and uses this identification to compensate for these effects by feedforward. Because these effects are identified as function of the states, a feedforward signal can be generated for arbitrary motions. The LFFC-scheme is depicted in figure 1.

The plant P, with transfer function $P(s)$, is assumed to be stable, linear and of the controllable canonical form. The (non-linear) state dependent effects, NL, act on the input of the plant and are assumed to be bounded and smooth. Commonly encountered effects are friction, cogging and mass deviation. The controller C, with transfer function $C(s)$, is assumed to be stable. The closed loop system is also assumed to be stable. The LFF-block is responsible for generating the feedforward signal on basis of the reference signal and is adapted on basis of the feedback signal. The signals encountered in this figure are r, the reference, e the error, d the so called disturbance signal. This disturbance signal originates from the state dependent effects. u_{ff} is the feedforward signal, u_{fb} the feedback signal, u the total control signal and u_{e} the excitation signal to the linear part of the plant.

The LFF comprises a function approximator that should learn the feedforward signal as a function of the states. This feedforward signal should contain the negate of the state dependent effects. Unknown parameters of the plant can be included in the block NL due to the assumption of the controllable canonical form. *By learning the negate of the signal d as function of its states, an approximation of the block NL is found.* This can be used to create a feedforward signal that should nullify these effects.

In the literature on LFF, the feedback control signal is commonly used as an approximation of the signal d, because the actual signal d cannot be measured. The transfer function from the signal d to the feedback signal u_{fb} can be calculated, omitting the arguments s for notational convenience, as:

$$\frac{u_{\mathrm{fb}}}{d} = -\frac{CP}{1+CP}. \tag{1}$$

In this calculation it is assumed that the disturbance signal is decoupled from the plant. This transfer function from the disturbance signal d to the feedback output is equal to one in the lower frequency range. However, for higher frequencies this transfer function will be smaller in absolute value and there will be a phase lag, if the plant and the controller are proper. When the phase shift of this transfer function becomes larger than 90° and the feedback signal is used as approximation of the signal d, it is shown in [12] that the system will become unstable. The instability is due to the phase shift between the actual signal d and the approximation of it by the feedback signal as illustrated in figure 2. In this figure x denotes some state on which the effect depends. The magnitude of the effect is given by u. The true relation between the state and the magnitude of the effect is given by the solid line. Suppose that the state is passed with a constant velocity, the dotted line shows the feedback control signal as function of the current state with a phase shift larger than 90°. From this figure it is clear that the phase shifted feedback signal is not a good measure of the actual disturbance signal.

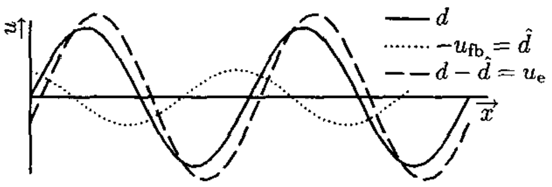

Figure 2: Instability due to phase shift

If the feedback signal is (partially) used for the feedforward signal in the next run, the sum of the disturbance signal and the feedforward signal gives the new excitation signal to the input of the linear part of the plant, u_{e}. This is shown in the figure with the dashed line. For stable learning, the combination of these should be smaller than the original disturbance signal, otherwise the excitation signal to the linear part of the plant will be larger as is the case in this example. This growth will continue the next iteration if the phase shift is again more than 90° and will cause instability.

2.2 Iterative Learning Control

When a task has to be performed repeatedly, and the starting state is the same at each starting time, then the tracking error will also be the same for every run, except for unreproducible disturbances like noise. Based on the tracking error, a feedforward control signal can be calculated that would compensate for the recorded tracking error. If this control signal is applied to the system in the next run, it will nullify the tracking error [7]. This control scheme is depicted in figure 3. In this figure u_k is the feedforward signal, e_k is the error signal, r is the reference signal and y_k is the output, all at run k. These signals are time signal, furthermore, for the reference signal it holds that $r(t) = r(t+T)$ in which T denotes the task execution time. As in figure 1, C is the controller, P is the plant and H_i are filters.

It is assumed that the controlled system is stable and that the plant is time invariant. Under these conditions, an update algorithm can be calculated for the feedforward signal that should nullify the error in the next run. The feedforward control signal for the next run for the scheme depicted in figure 3

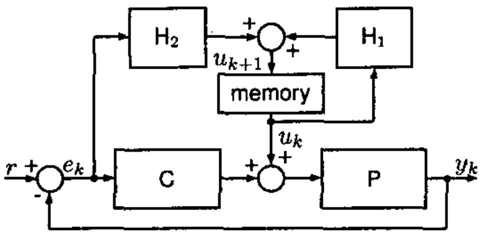

Figure 3: ILC-setting

is given as:

$$u_{k+1} = H_1 u_k + H_2 e_k. \tag{2}$$

The transfer function from the feedforward signal to the error is:

$$U = -\frac{e_k}{u_k} = \frac{P}{1+CP}. \tag{3}$$

Substituting this in the update equation (2), results in:

$$\begin{aligned} e_{k+1} &= -\frac{P}{1+CP} u_{k+1} \\ &= -\frac{P}{1+CP} H_1 u_k - \frac{P}{1+CP} H_2 e_k \\ &= \left(H_1 - H_2 \frac{P}{1+CP} \right) e_k. \end{aligned} \tag{4}$$

The error will converge if the next error is smaller than or equal to the present, which results in [8]:

$$\left\| \left(H_1 - H_2 \frac{P}{1+CP} \right) \right\|_\infty \leq 1 \tag{5}$$

Next to the convergence, the asymptotic error is of importance. The asymptotic error is shown to be [8]:

$$e^\infty = \left(\frac{1 - H_1}{1 - H_1 + H_2 \frac{P}{1+CP}} \right) e^0 \tag{6}$$

Based on (5) and (6) the following filters will be used for H_i:

$$H_1 = 1 \qquad H_2 = \left(\frac{P}{1+CP} \right)^{-1} = U^{-1} \tag{7}$$

The following should be noted on H_2:

- If $U(s)$ contains zeros in the right half plane, then its inverse, $H_2(s)$, will have poles in the right half plane and will be unstable.
- Since $P(s)$ is usually proper as well as $C(s)$, $U(s)$ is proper. Therefore $H_2(s)$ is not proper, resulting in amplification of high frequencies.
- $U(s)$ is uncertain, especially for higher frequencies, which makes $H_2(s)$ uncertain.

The first of these items can be coped with by the Zero Phase Error Tracking Control (ZPETC) method of [11]. This method finds a stable inverse of a discrete plant, and compensates for the phase shift that is induced by the uncancelled zeros. The cancelling of the phase is important because this causes that the next feedforward control signal will be applied at the correct time. This importance is furthermore stretched in [3]. The result of the ZPETC algorithm on a proper plant will result in a transfer function with negative delay. This is not a problem for ILC, because the filtering is done off-line.

The last two problems can be solved by including a low-pass filter, Q, in the learning filter. The new update rule becomes:

$$u_{k+1} = Q\left(H_1 u_k + H_2 e_k\right), \tag{8}$$

by which the condition for stability (5) becomes:

$$\left\| Q \left(H_1 - H_2 \frac{P}{1+CP} \right) \right\|_\infty \leq 1 \tag{9}$$

The inclusion of this low-pass filter is needed to guarantee stability, but it degenerates the performance. Fast alterations required to compensate for the state dependent effects cannot realised anymore, therewith making it impossible to nullify these fast changing effect.

3 PHASE CORRECTED LFFC

The transfer function from the disturbance signal to the error signal is given as:

$$\frac{e}{d} = -\frac{P}{1+CP}.$$

which is equal to (3). The inverse of this transfer function gives the disturbance signal based on the error signal. So, instead of using the feedback control signal as an estimation of the disturbance signal, the filtered error signal can be used to obtain an disturbance estimate. This estimate will not contain the phase error. The transfer function of this filter is identical to the learning filter in the ILC case, and therefore the same methodology given in section 2.2 can be used to obtain a stable inverse. This control scheme is depicted in figure 4. In this scheme the learning filter in combination with some low-pass filter is denoted as Q₁L. The filter Q₂ is a second low-pass filter that is used to filter the output of the function approximator. In this figure the dashed lines represent the adaptation signals, while the solid lines represent the control loop.

The transfer function of L is the stable inverse of the U as given in (7) and can be computed by the ZPETC algorithm. This learning filter requires information of the future, because it comprises the inverse of a time delay. In the off-line ILC setting, this might not pose a problem, however in on-line control this information is not present. *Hence, it is not feasible to apply ZPETC to learn for the current command state, however, instead of trying to update the approximator for the current command state, it* is *possible to update the feedforward signal for several steps in the past. Because for that sample, several steps in the future* are *known*. This makes it possible to implement the phase correction.

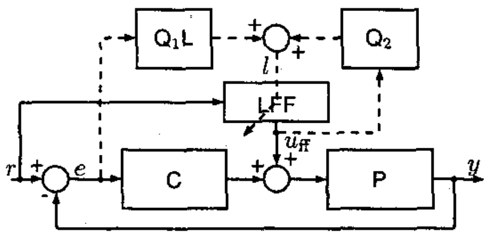

Figure 4: The LFFC-scheme with phase correction

3.1 Some notes on Stability

This section will give some notes on the stability to get insight in the these issues for phase corrected LFFC; a mathematical proof is not presented.

The difference between the LFFC and the ILC scheme to compensate for state dependent effects, if the task is repetitive, is concentrated on how the information is stored. In the LFFC scheme the information is stored as a functions of the states, while in the ILC scheme the information is stored as function of the time. If the function approximator is capable of storing the required feedforward signal perfectly, the generated feedforward signal of ILC and LFFC are identical. This means that the stability condition of ILC, (eq. 5), has to hold for both schemes.

In ILC the feedforward control signal is computed for the next run. This means that the output of the the memory will not influence the error this run. Due to this, the loop of the error to the memory will not contain dynamics that can destabilise the controlled system. However, LFFC is not restricted to repetitive motion tasks, and therefore the update of the approximator will introduce dynamics through the adaption of the function approximator. The effect of the alteration can be made so small, by adapting the function approximator only a small part in the direction of the update signal, that the dynamics of the approximator are neglectable. The small adaptation makes the function approximator quasi-static and therefore will not cause instability.

3.2 Some notes on Performance

The steepness of the state dependent effect plays an important role in the final tracking error. This is due to two factors:

1. Learning at commanded states

2. Low-pass filter Q

The effects are told to be learnt as function of the states. However, not all the required states can generally be measured, and those that can be measured will contain noise. If states are estimated from the noisy measurements, these calculated states can contain considerable uncertainty, especially if differentiation is needed. Therefore, LFFC uses not the actual states, but the commanded states. These commanded states are present as by-product of the reference signal, and are therefore noise free. However, there will be some deviation between the actual states and the commanded states. This will limit the performance, because a feedforward signal is generated for an incorrect state. The difference between the feedforward signal and the actual state dependent effect, and relative to this the tracking error, will dependent on the steepness of the state dependent effect.

Furthermore, due to the low-pass filters Q_i, the learning signal will not be infinitely steep, and is therefore not be able to compensate for fast (seemingly discontinuous) effects.

4 VALIDATION

Both simulations and experiments are used to validate the phase correction. Simulations are used, because they make it possible to focus on a sole phenomena while this might be impossible by experiments. Experiments are performed to show that the tracking error will decrease under real circumstances. The setup that will be used in both simulations and experiments is a linear motor.

4.1 Simulations

To show the effect of the phase correction two sets of simulations are performed:

1. Test the influence of the phase correction on the learning.

2. Test the effect of the steepness of the non-linear effect.

The simulations will be performed with a model of a synchronous linear motor. The dominant linear behaviour is that of moving mass. On the input-force of the moving mass two non-linear state dependent effects act. These effects are friction and cogging. The equation of this system is given as:

$$m\ddot{x} + F_{\text{fric}}(\dot{x}) + F_{\text{cog}}(x) = F_{\text{ext}} \tag{10}$$

in which m denotes the mass, $F_{\text{fric}}(\dot{x})$ the friction force, $F_{\text{cog}}(x)$ the cogging force and F_{ext} the externally applied force. The cogging and the friction that are used in the model, are measured values of these effects on the real setup. The used feedback controller is a PD-controller which has been tuned to give realistic performance with the plant. Because the final setup will be partly continuous, the motor, and partly discrete, the controller, this configuration is also mimicked in the simulation.

The function approximator that is used in the simulations and the experiment is a 2^{nd}-order B-Spline Network [1]. This function approximator can be seen as a multidimensional interpolation table. To obtain the output for an input that is not specified in the table, the neighbour values are interpolated.

The learning filter L is chosen as the inverse of the discretised transfer function U. The final filter equals the filter as given in [3]. The low-pass filter is a FIR-filter by which the coefficients are calculated to minimise the least squares criterion for a low-pass filter [10]. This results in the following coefficients:

$$b(n) = \frac{\sin(\omega_c T_s (n - N/2))}{\pi(n - N/2)} \tag{11}$$

in which the n is the coefficient number, N the order of the filter, ω_c the cutoff frequency and T_s the sample time.

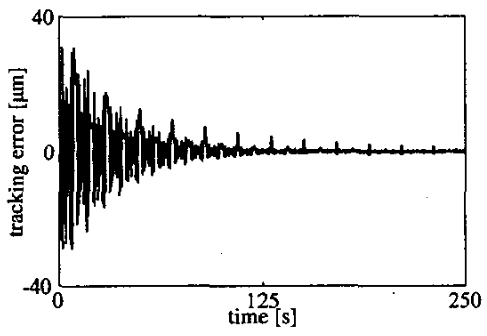

Figure 5: Tracking error for the slow motion. Uncorrected phase.

4.1.1 Phase influence

To test the influence of the phase correction, one non-linear effect is assumed to be present, namely the cogging. Cogging is approximately a sine function that depends on the position. To clearly demonstrate the effect of the phase on the learning, the motion that is performed during learning is done first slow and later on the same motion is performed fast. The slow motion will inject a signal d within the bandwidth resulting in a good approximation of this signal by the feedback signal, while for the fast motion, this will not be the case. The motion is to the end of the motor and back again.

In figure 5 the tracking error for the phase uncorrected scheme is given. In figure 6 the estimation of the disturbance signal in the phase uncorrected scheme is shown as function of the position. The phase lag of $\frac{u_{fb}}{d}$ is 0.5° for the dominant injected frequency. Due to the small phase shift, the feedback signal is a good estimate of the disturbance signal and the learning of the feedback signal as function of the position can compensate well for the cogging. The phase corrected scheme is not shown, because it is identical to the given figures.

In figure 7 the estimation of the disturbance signal for the phase uncorrected scheme is given as function of its state for the fast motion. The phase lag for the dominant injected frequency is approx. 75° and the feedback signal is a bad representation of the disturbance signal, making it nearly impossible to identify the cogging based on these signals. For the fast motion, stable behaviour could not be obtained with phase uncorrected learning. The phase corrected scheme gave results equal to those of figure 5 and 6. The phase corrected learning method did not have any problems with the fast movements and could therefore compensate for the cogging in both simulations.

4.1.2 Steepness

The final tracking error of the learning scheme is influenced by the steepness of the state dependent effect. To emphasise the influence of the steepness, a simulation is performed with only one steep state dependent effect. This effect is given by $f(x) = A\tanh(s(x - x_0))$ which depends on the state x, which has a steepness parameter s, passes through zero at x_0 and has an amplitude of A.

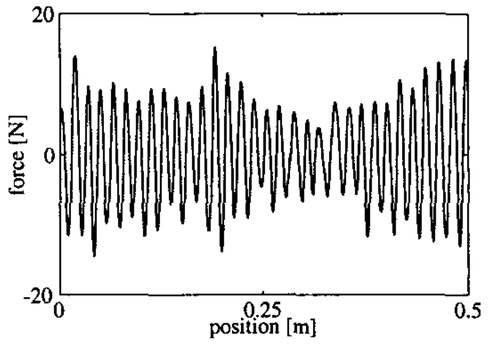

Figure 6: The disturbance estimation for the slow motion.

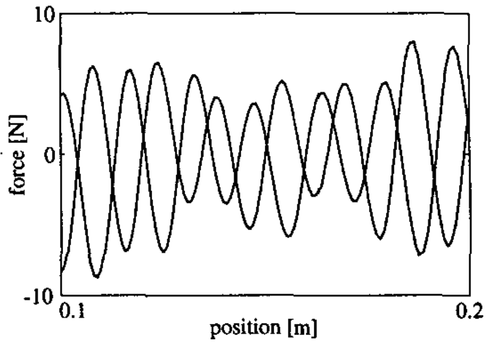

Figure 7: The disturbance estimation for the fast motion.

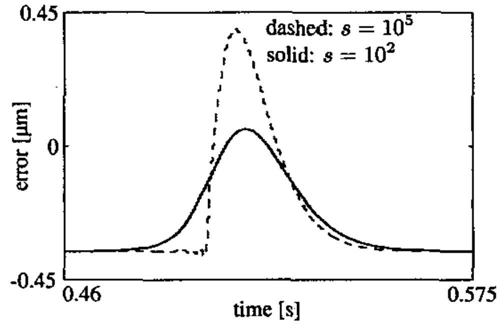

Figure 8: Remaining tracking error due to steep function

The tracking error after learning of the phase corrected method is given in figure 8. In this figure the solid line is for $s = 10^2$ while the dashed line is for $s = 10^5$.

It can be observed that a tracking error will remain for steep functions. The tracking error given in figure 8 illustrates that a steeper function will give a larger final tracking error. This is due to two factors: first the deviation between the true states and the command states and second due to the low pass filter Q.

4.2 Experiment

The actual linear motor exhibits, next to friction and cogging, position and velocity dependent friction as well. Furthermore, the linear part of the motor contains vibration modes starting at about 40 Hz that were not included in the model. The motor is controlled by a PD-controller, running

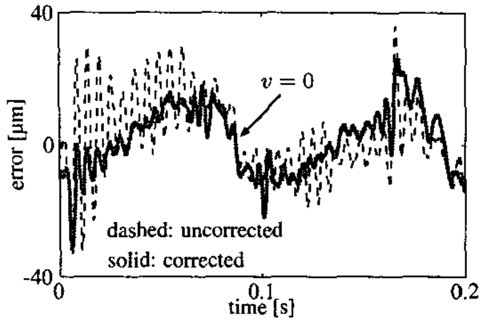

Figure 9: Comparison of tracking error for corrected and uncorrected learning on the true setup.

at a frequency of 2 kHz. The bandwidth of the controller and the low-pass filter Q were tuned such that system performed well. The learning rate was tuned such that the system remained stable. The B-Spline network that is used in the experiment is a parsimonious network [2]. This means that not one two-dimensional network is used, but two one-dimensional networks. The first network learns the effects as function of the velocity, while the second network learns the effects as function of the position. Although we known that velocity-position dependent friction is present, this parsimonious setup is used, because the effect of the velocity-dependent friction is small relative to the velocity dependent friction and the position dependent cogging. Furthermore, by using more parameters the variance on these parameters will increase [4].

In figure 9 the tracking error of both the phase uncorrected and phase corrected LFFC schemes are shown. In this figure the dashed line is the uncorrected LFFC, while the solid line is the corrected LFFC. The offered motion profile was a second order path with an acceleration of 5 m/s^2, a velocity of 0.9 m/s and a stroke of 0.5 m. The tracking error of the phase corrected method is clearly smaller in this case than of the phase uncorrected method. Due to the introduced phase shift, the feedback signal is unable to learn the cogging. The learning rate of the uncorrected method had to remain lower in order to keep the scheme stable. In this figure it can be observed that the tracking error is rather large if the velocity changes sign. This is due to the friction which has a discontinuity around zero. This makes it impossible to learn perfectly as seen in subsection 3.2. The remaining error can be explained by the position dependent friction as well as the presence of higher order dynamics that were not incorporated in the design of the learning filter L.

5 CONCLUSION

In this paper a phase correction is introduced for Learning Feedforward Control. This phase correction decreased the tracking error relative to ordinary LFFC as shown with simulations as well as with experiments. This phase correction makes use of the ideas of Iterative Learning Control and the Zero Phase Error Tracking Method. Although these methods require to have knowledge of the future, this is not required if the update of the learning feedforward element is postponed for this time. Furthermore, during the experiments it was found that the learning rate could be larger then for phase uncorrected LFFC, resulting in faster learning.

REFERENCES

[1] M. Brown and C. Harris. *Neurofuzzy adaptive modeling and control.* Prentice Hall International, UK, 1994.

[2] T.J.A. de Vries, W.J.R. Velthuis, and L.J. Idema. Application of parsimonious learning feedforward control to mechatronic systems. *IEE Proc. D: Control Theory & Applications*, 148(4):318–322, July 2001.

[3] J. Van Dijk, R. Tinsel, and E. Schrijver. Learning control of direct drive systems with cogging force disturbances. In B. Samali, editor, *Fifth Motion and Vibration Conference, MOVIC2000*, pages 381–386, Sydney, Australia, December 2000.

[4] N.R. Draper and H. Smith. *Applied Regression Analysis.* Wiley Series in Probability and Statistics. John Wiley & Sons, Inc., New York, 3rd edition, 1998.

[5] M. Kawato, K. Furukawa, and R. Suzuki. A hierarchical neural network model for control and learning of voluntary movement. *Biological Cybernetics*, 57:169–185, 1987.

[6] B.J. de Kruif and T.J.A. de Vries. Improving price/performance ratio of a linear motor by means of learning control. In *Proceedings of the IFAC Mechatronics 2002*, Berkley, USA, 2002.

[7] R.W. Longman. *Iterative Learning Control: Analysis, Design, Integration and Applications*, chapter 7: Designing Iterative Learning and Repetitive Controllers, pages 107–146. Kluwer Academic Publishers, 1998.

[8] K.L. Moore. *Iterative Learning Control for Deterministic Systems.* Advances in Industrial Control Series. Springer, Londen, UK, 1992.

[9] G. Otten, T.J.A. de Vries, J. van Amerongen, A.M. Rankers, and E.W. Gaal. Linear motor motion control using a learning feedforward controller. *IEEE/ASME Trans. Mechatronics*, 2(3):179–187, 1997.

[10] D. Schlichthärle. *Digital Filters, Basics and Design.* Springer, New York, 2000.

[11] M. Tomizuka. Zero phase error tracking algorithm for digital control. *Journal of Dynamic Systems, Measurement, and Control*, 109:65–68, March 1987.

[12] W.J.R. Velthuis. *Learning Feed-Forward Control, Theory, Design and Applications.* PhD thesis, University of Twente, Enschede, Febuary 2000.