Linear Motor Motion Control Using a Learning Feedforward Controller

Gerco Otten, Theo J.A. de Vries, *Member, IEEE*, Job van Amerongen, *Member, IEEE*, Adrian M. Rankers, and Erik W. Gaal, *Member, IEEE*

***Abstract*—The design and realization of an on-line learning motion controller for a linear motor is presented, and its usefulness is evaluated. The controller consists of two components: 1) a model-based feedback component and 2) a learning feedforward component. The feedback component is designed on basis of a simple second order linear model, which is known to have structural errors. In the design, emphasis is placed on robustness. The learning feedforward component is a neural-network-based controller, comprised of a one-hidden-layer structure with second-order *B*-spline basis functions. Simulations and experimental evaluations show that, with little effort, a high-performance motion system can be obtained with this approach.**

***Index terms*– Intelligent control, learning control systems, linear synchronous motors, mechatronics, motion control, neural network applications**

I. Introduction

Modern laser cutting, high-speed milling and scanning machines require fast and/or accurate linear motions. Linear electromotors are becoming increasingly popular in such applications. Compared to traditional drives that use rotational electromotors and lead screw or toothed belts (indirect-drive), the direct-drive linear motor exhibits the property of contactless transfer of electrical power to translational mechanical power according to the laws of magnetic induction. The electromagnetic force is applied directly to the payload without the intervention of a mechanical transmission. For linear motion actuation, the linear motor design, therefore, features the following advantages over its traditional rotary-motor and transmission counterpart [1]:

- less friction and no backlash, resulting in high accuracy;
- no mechanical limitations on acceleration and velocity; (the velocity is only limited by the bandwidth of the position measurement system (usually an encoder) or by the power electronics);
- higher reliability and longer lifetime, due to mechanical simplicity.

Hence, the performance of such a motion system is hardly limited by mechanical elements if the remainder of the construction (such as the supporting frame) is designed sufficiently stiff.

In this paper, we consider a specific type of linear electromotor, namely, a brushless permanent-magnet motor. This type of motor is attractive because it offers high-power density, reliability, and efficiency. However, it suffers from a positional dependency in the thrust force, herein called the force ripple. Due to the direct drive principle of a linear motor, the force ripple can have a significant effect on the positional accuracy at the load. As the force ripple is a disturbance that is *a-priori unpredictable, yet highly reproducible*, one can try to minimize it by adjusting the feed currents for the motor *on line* using a learning controller [2]. This idea will be pursued here as well, however, with the aim to control motion rather than force [3]. Hence, we want to eliminate positional inaccuracy due to force ripple or any other (reproducible, slowly varying) disturbance. For this purpose we consider a *learning feedforward controller* structure [4], i.e., a controller consisting of a model-based feedback component and a learning feedforward component.

We first discuss the linear motor motion system in more detail in Section II. Next, the design of the learning control system is discussed (Section III). Simulations and experimental results will be presented in Section IV. We end with conclusions in Section V.

II. Linear Motor

A. Working principle

In Fig. 1 the working principle of the linear motor being studied is depicted. It is comprised of two main parts:

- a number of base-mounted permanent magnets forming the stator;
- a translator (as counterpart of the rotor in a rotating motor) formed by a number of iron-core coils.

By applying a three-phase current to three adjoining coils of the translator, a sequence of attracting and repelling forces between the poles and the permanent magnets will be generated. This results in a thrust force experienced by the translator. Basically, the motor is a synchronous permanent-magnet motor with electronic commutation. The interested reader is referred to [5] and [6] for literature on linear motors.

Manuscript received August 19, 1996; revised March 24, 1997. This work was supported by Philips' CFT. Recommended by Technical Editor K. Ohnishi.

G. Otten is with Hollandse Signaalapparaten, Hengelo, The Netherlands.

T.J.A. de Vries and J. van Amerongen are with the Control Laboratory, University of Twente, P.O.Box 217, 7500 AE Enschede, The Netherlands (e-mail: icontrol@rt.el.utwente.nl).

A.M. Rankers and E.W. Gaal are with the Centre for Manufacuting Technology, Philips Electronics N.V., Eindhoven, The Netherlands.

Publisher Item Identifier S 1083-4435(97)06982-2.

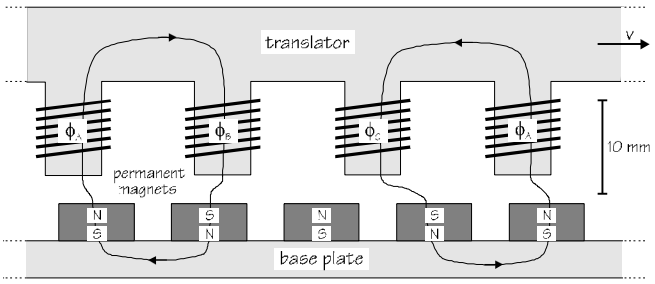

Fig. 1. Working principle of a (three-phase) synchronous permanent-magnet linear motor.

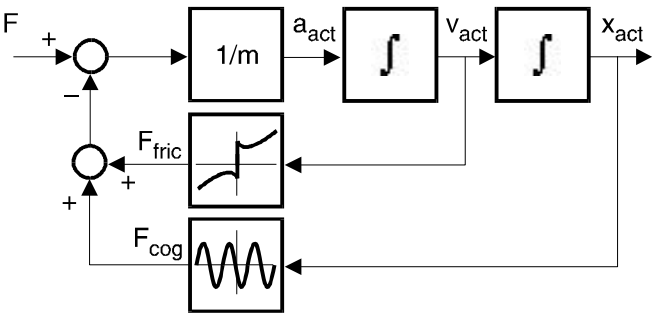

Fig. 2. Nonlinear block diagram model of the linear motor.

In addition to the thrust force, other forces are generated in the synchronous permanent-magnet motor by two physical phenomena [5].

1) *Cogging or detent force* – The translator of the linear motor consists of iron-core coils. The attraction between the permanent magnets and the iron cores causes a force in direction of motion. This force depends only on the relative position of the motor coils with respect to the magnets and is always present, even when there is no current flowing in the motor coils.
2) *Reluctance force* – When the position of the translator changes, the winding self-inductance varies. When current flows through the coils, this causes a position-dependent force in direction of motion.

All of these force sources are greatly affected by the magnetic structure of the motor. Ideally, the motor thrust force is position independent. Cogging and reluctance force together cause an undesirable positional dependency in the thrust force, i.e., the force ripple.

B. Force-Ripple compensation

By properly designing and realizing the spatial layout of the permanent magnets and the feed currents of the motor, the force ripple can be minimized. However, to reduce manufac-turing costs of the motor, it is desirable to have not-too-tight tolerances on both magnetic properties and placement accuracy of the perma-nent magnets. One way to allow this and yet minimize force ripple due to lack of spatial symmetries is by constructional measures.

- Use noniron cores instead of iron cores for the translator windings.
- Skew the stator magnets relative to the direction of movement of the translator.

Unfortunately, these measures reduce the maximum force that can be generated and the efficiency of the motor. One can instead compensate the force ripple in feedback by modifying the feed currents with on-line evaluations of off-line computed expressions in terms of the back EMF [7], [8]. The merits of this approach are that one obtains better positional accuracy without sacrificing maximum force and motor efficiency. Recently, it has been shown that it is beneficial to determine the required current profile on-line using adaptation, as this allows to compensate for nonlinearities and for effects particular to the motor under control [2]. As an alternative to feedback compensation using measurements of back EMF, one can use feedforward compensation on basis of an experimentally identified first-order approximation of the force ripple [9]. This yields good results and does not require measurement of back EMF, which is attractive. Here, we pursue the combination of the latter ideas, i.e., we try to compensate the force ripple using learning (adaptive) feedforward control. Therefore, we need no measurement of back EMF and may be able to compensate undesirable effects particular to the motor under control, without having to identify a low-order approximation of the main disturbances.

C. Linear motor specifications

The linear motor considered here is a current-controlled three-phase motor driving a carriage supported by a number of recirculating ball bearings. The motor uses sinusoidal commutation. The position of the carriage is measured by an incremental linear encoder with a measurement resolution of 0.5 *m*m. The carriage is comprised of, among other things, the translator of the motor and an additional dummy mass to realize a total carriage mass of 37 kg, resembling a practical setup. As the ball bearings are not perfect, the system is not free of friction.

Fig. 2 depicts a block diagram model of the motor. This non-linear model is used as starting point for learning controller design. It comprises two nonlinear blocks.

- The friction is modeled with a combination of Coulomb friction, viscous friction and a so called Stribeck effect [10], which can be interpreted as stiction.
- The force ripple is described by a sinusoidal function of the load position.

The linear motor under concern is applied in, among other things, scanning machines and pick and place machines, where high positioning accuracy is required. The main motor characteristics are: F_{max} = 750 N, $m_{translator}$ = 7 kg, m_{load} = 30 kg, v_{max} = 2 m/s, a_{max} = 20 m/s^2. In a scanning machine, movements take place with low velocities (up to approximately 50 mm/s) and low accelerations. In this case, cogging and, to a lesser extent, friction disturbances are the dominant causes for position errors of the translator and, accordingly, the payload. In a pick and place machine, translator movements take place with high velocities (up to 2 m/s) and large accelerations (up to 20 m/s^2). In such applications, the influence of the cogging disturbance is relatively small; the influence of dynamic

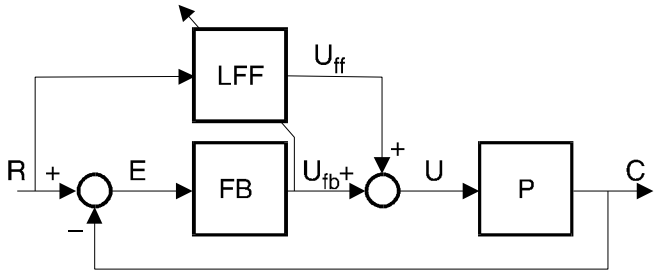

Fig. 3. Learning feedforward controller.

(acceleration and deceleration) effects is more dominant (see Section IV-B).

III. CONTROLLER DESIGN

Industrial mechatronic servo systems are mostly controlled by a proportional-plus-derivative action (PD)-type feedback controller in combination with a feedforward controller and disturbance compensation (*I*-action). For good system performance, the feedforward controller is provided with the inverse of the *a priori* known process model. This implies that modeling and identification procedures are essential for obtaining controllers of sufficient quality. In other words, knowledge of the process is a prerequisite for good control. Often, the available knowledge of the process to be controlled is not appropriate for controller design. In this case, for example, the main disturbances, friction and force ripple, are not known quantitatively beforehand and, hence, cannot be compensated for properly. In such a situation it makes sense to use available knowledge of the process during controller design, as far as viable, and to acquire additional process knowledge on-line (during control) for additional performance. Such an approach can be implemented by using a learning controller that is comprised of the following two components (Fig. 3) [4]:

- a feedback component (*C*), designed on basis of the *a priori* available process model, with the aim of delivering a robust controlled system with large stability margins;
- a separate learning component (*Q*), equipped to acquire and utilize process knowledge which is not taken into account (quantitatively) in feedback design, so that the system performance is optimized during control.

So the design of the learning controller can be divided into two steps: 1) feedback component design and 2) feedforward component design. Before discussing the latter, we will elaborate upon the concept behind the learning feedforward controller.

A. Feedback component

Because the friction and ripple characteristics, shown in Fig. 2, are not known accurately, we do not take them into account in feedback controller design. The resulting model of the linear motor is a simple second order system (moving mass):

$$\ddot{x} = \frac{1}{m} F_{\text{thrust}} \qquad (1)$$

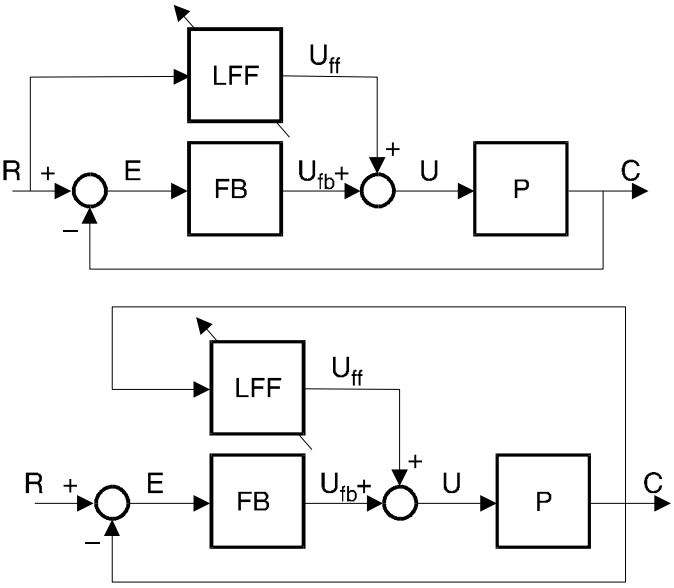

Fig. 4. Learning (upper) in feedforward versus (lower) in feedback.

The feedback controller has been designed considering the model of (1). As the errors of the model-based controller will be reduced by the learning component, the demands on feedback tracking performance are not strict. For this reason, the feedback parameters have been chosen sufficiently small, so as to obtain a safe stability margin. A PID-type feedback controller is used during the experiments. The parameters of the PID-type feedback controller are adjusted according to some tuning rules, with very little effort. The most important goal of the feedback controller is to realize a stable closed loop system. A PID controller with K_p = $2.8{\cdot}10^5$ N/m, K_d = $5.5{\cdot}10^3$ Ns/m and K_i = $7.2{\cdot}10^6$ N/ms (m = 37 kg) result in a well damped and stable closed loop system with a system bandwidth of approximately 25 Hz.

B. Learning feedforward concept

First, locating the learning component in the feedforward rather than feedback loop will be motivated. Then, the choice of the learning network, as it is implemented as a neural network, is motivated. The chosen network learning rule will be discussed next.

1) *Learning in Feedforward Versus in Feedback:* In a tracking control system, the learning component can be located either in the feedback loop or outside the feedback loop, resulting in feedforward learning (Fig. 4).

One reason for choosing the feedforward loop is that, for reproducible disturbances such as unmodelled process dynamics, feedforward compensation is principally faster than feedback compensation. In other words, we get the best performance if we try to learn the inverse process in the feedforward path.

A second reason follows from considering stability at an intuitive level. If we disregard the learning loop, the transfer functions of the closed-loop system with a learning feedfor-ward and a learning feedback controller are, respectively,

$$\frac{Y}{R} = \frac{Q_{\text{ff}}(s)P(s) + C(s)P(s)}{1 + C(s)P(s)} \qquad (2)$$

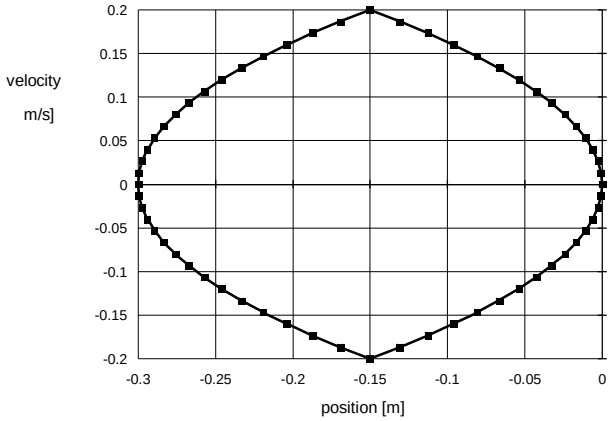

Fig. 5. Linear motor motion as function of network inputs (low-velocity movement).

$$\frac{Y}{R} = \frac{C(s)P(s)}{1 - Q_{\text{fb}}(s)P(s) + C(s)P(s)} \qquad (3)$$

As the stability of the controlled system is determined by the feedback loop [see Fig. 4 and (2) and (3)], feedback learning may endanger stability. Stable feedback learning or adaptation mechanisms can be constructed, but require detailed analysis of stability robustness. In feedforward learning, this seems less necessary.

However, although not explicit in Fig. 4, the feedforward learning structure also incorporates a contribution to the feedback loop, caused by the fact that any learning signal will be based on measurements of process output. We come back to this later, when choosing the learning rule.

2) Choice of learning network: The learning feedforward controller has to create a non-linear mapping between the reference input(s) and the force output. This mapping has to be learned by the controller. Optional realizations are an adjustable look-up table, a multilayer perceptron [11], a radial basis function network [12] or a single-layer *B*-spline network [13]. In line with previous research [4], we have chosen to use *B*-spline networks. Such networks are comparable to radial basis function networks; they are one-hidden-layer networks with adaptable weights between the hidden layer and output layer only and, in case of multiple input variables, tensor product construction. They make use of *B*-spline basis functions that unequal zero on a restricted part of the input domain only. Hence, they feature a relatively short evaluation time for reading their content and for learning.

3) Learning rule: The standard learning rule used with a *B*-spline network is based on the back propagation learning rule [11]. Changing the values of the network weights in the direction of steepest descent with respect to error results in minimizing the summed squared error of the network. Consider an input $\boldsymbol{x}$ together with a desired output $y_t(\mathbf{x})$. When the weights are adjusted according to the following learning rule, the actual output will approach the desired output.

$$w_{i_1 \cdots i_k,\,\text{new}} = w_{i_1 \cdots i_k,\,\text{old}} + g \cdot \{y_t(\boldsymbol{x}) - y(\boldsymbol{x})\} \cdot b_{i_1 \cdots i_k}(\boldsymbol{x}) \qquad (4)$$

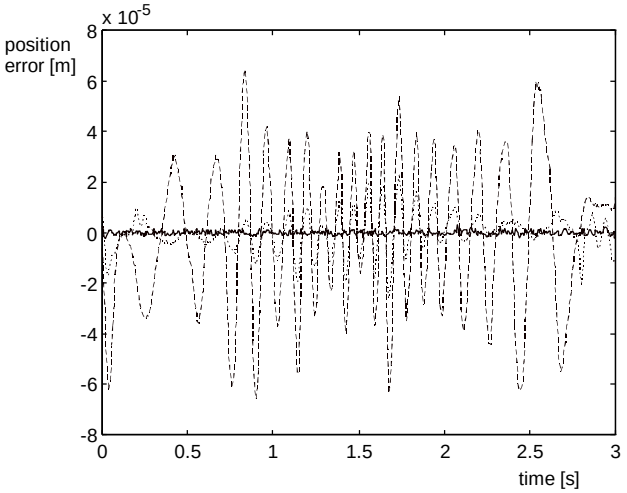

Fig. 6. Position error before learning (dashed) and during second (dotted) and tenth (solic) runs while learning.

with learning rate g satisfying $0 < g < 1$ and (k-dimensional) basis functions $b_{i_1 \cdots i_k}(\boldsymbol{x})$.

In the ideal case, the feedforward component should contain the mapping from the reference signals to corresponding control signal, such that no position error occurs by applying this control signal to the process. The total feedforward structure equals the inverse process model in that case. As long as this mapping is not perfect, position errors will occur which will be compensated for by the output of the feedback component. Hence, we can interpret the feedback steering as an error measure for the feedforward steering. Therefore, using the feedback control signals as the output error measure for the learning feedforward controller is justified, and we obtain the structure of Fig. 3. This changes the learning rule into:

$$w_{i_1 \cdots i_k,\,new} = w_{i_1 \cdots i_k,\,old} + g \cdot u_{fb}(\boldsymbol{x}) \cdot b_{i_1 \cdots i_k}(\boldsymbol{x}) \qquad (5)$$

With (5), it becomes clear how the "feedforward component" is incorporated in a feedback loop due to learning. Hence, the addition of a learning feedforward component may endanger system stability. Further research regarding this is reported in [14]; at this moment, we assume that, as long as g small, the effect of learning on stability can be neglected. An additional benefit of a small learning rate is that the learning behaviour becomes robust for a considerable amount of sensor noise [4].

C. Learning feedforward component design

A number of design choices remain to be made. The most important ones are the following.

1) Network inputs: As a linear motor mainly suffers from position-dependent disturbances (cogging), it seems wise to take at least the setpoint position as network input. Exper-iments are performed with setpoint velocity as additional network input. Fig. 5 shows that for the motion profiles to be used in the experiments (see the Appendix), we have obtained a unique combination of network inputs during the time span of the motion in this way.

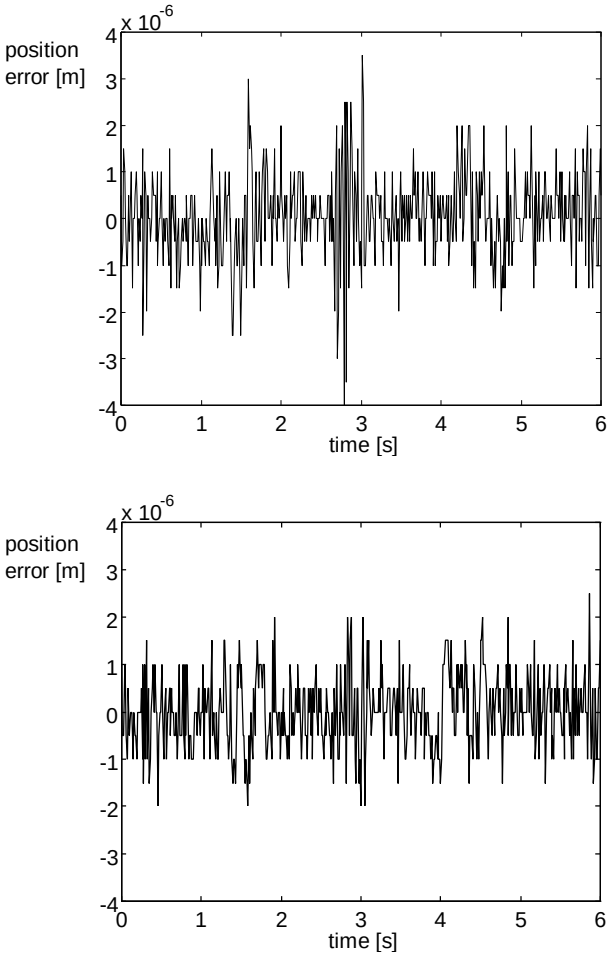

Fig. 7. Position error during (upper) tenth run and (lower) twentieth run while learning.

This unique codification is essential because disturbances not depending on position or velocity (for example acceleration disturbances) are allocated to a combination of these network inputs. Proper steering signals to compensate for this cannot be stored in the network if the codification is not unique during the time span of the motion, for example, if only setpoint position was taken as network input.

2) Order of Spline Functions: To obtain continuous control signals with bounded time derivatives, at least second-order splines are required. Higher order interpolation and, hence, additional smoothness can be obtained at the cost of extra computational effort. We have chosen to experiment with second-order spline functions.

3) Learning Rate: A learning rate g=0.1 is used. This learning rate is chosen as a compromise between fast learning and assuring stability. From simulations and experiments this turned out to be a proper value.

4) Number of Splines and Distribution of Splines on Input Space: The *B*-splines are placed on the input spaces according to a predefined grid for each input space. We prefer to use a uniform distribution as much as possible, for computational efficiency.

In our experiments, the position of the carriage varied in the range [–0.4m … 0.1m]. Over this range, 300 splines were

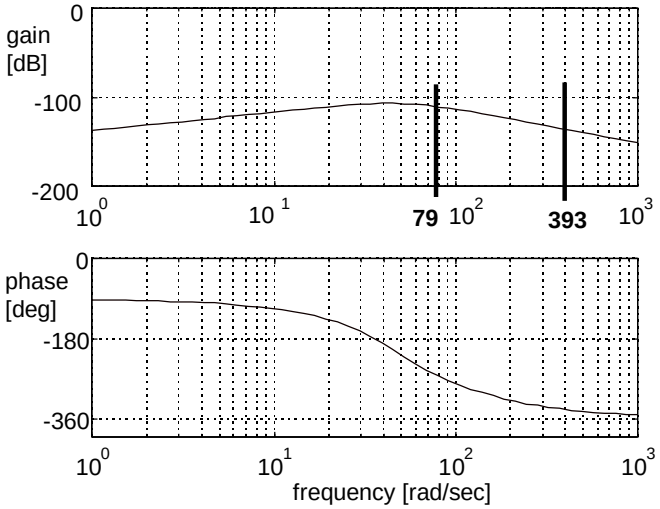

Fig. 8. Bode diagram of disturbance suppression of cogging.

distributed uniformly. Variation of the feedforward steering signal over the position input was expected to be mainly determined by cogging. With the given amount of splines, these variations can be well approximated.

Over the velocity input, variations in the feedforward steer-ing signal were expected to be mainly determined by friction. From the friction characteristic assumed in Fig. 2 we know that it is discontinuous at zero velocity (Coulomb friction) and a strongly non-linear friction force variation around this velocity due to Stribeck curve effects. To allow the network to approximate this, two measures were taken. Firstly, the velocity input domain was separated into two parts, one for negative velocity and one for positive. Secondly, the spline density was chosen higher, close to zero: 3 in the velocity range [0.05 … 1.0 m/s] (both directions); and 9 in the range [0.0 … 0.05 m/s] (again both directions).

The memory requirement of the network is determined by the number of *B*-splines used. Approximately 100 kb of memory were available for the learning feedforward control algorithm in the linear motor setup. The total size of the network is 300·(14+14) = 8400 spline functions. A 32 bit floating point number representation was used during implementation, so that 8400·4=33600 bytes of memory were used for storage of the network weights.

IV. RESULTS

During simulations and experiments, the linear motor was required to let the carriage follow a second order profile for the position.

A. Simulations

Simulations were performed with the simulation program *20-sim*® [15] to demonstrate feasibility of learning control. A non-linear motor simulation model was obtained by extending (1) with friction and cogging effects, as depicted in the motor model of Fig. 2. The friction force was assumed to consist of viscous friction (linear in the velocity), Coulomb friction (dependent of the sign of the velocity) and Stribeck curve effects (variation in

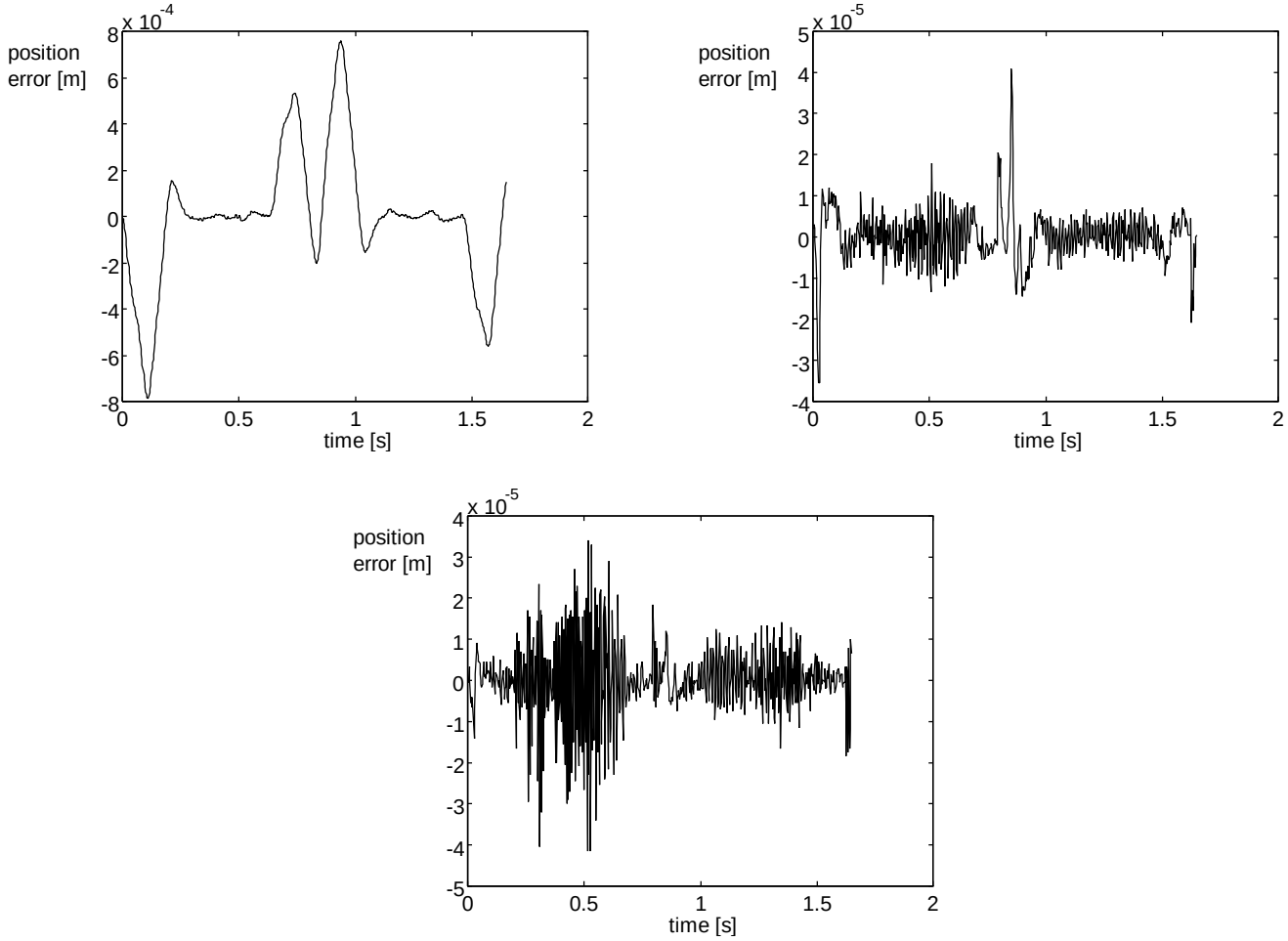

Fig. 9. Unstable learning process: position error (upper left) before learning and (upper right) during tenth and (lower) twentieth runs.

friction coefficient at low speed). The cogging characteristic was assumed to be sinusoidal, with a spatial period of 16 mm and an amplitude of 8.5 N. In reality, the cogging disturbance is more complex in shape (e.g., due to variations in magnet dimensions), but has a comparable period and amplitude. No parameter mismatches were introduced into the simulation model compared to the model used for feedback controller design.

The simulation results indicated that the proposed learning controller is able to compensate the effects of friction, cogging and feedback controller imperfections [16].

B. Experiments

Experiments were performed with the actual linear motor setup as well. The control system was implemented using a dSPACE® DS1102 controller board with a TMS320C31 40 MHz floating-point digital signal processor (DSP). The controller executed at a sampling frequency of 1.7 kHz. The network weights were initialized to zero before the experiments were performed. The main results for both high and low velocity trails are shown here; for a more complete and detailed description of the experiments performed with the linear motor system, the reader is referred to [16].

1) Low Velocities: The required motion was a movement back and forth over 0.3 m within 6 s. (see the Appendix). This motion was covered repeatedly, and Fig. 6 shows the position error of the carriage before learning, during the second and tenth run while learning. The position error signals are shown for the first half of the movement only; the second half gives similar results.

The learning behavior of the *B*-spline network during repeated coverage of the specified path is obvious. These experimental results show that the learning feedforward controller actually reduces the position errors of the linear motor system drastically. The position error of the carriage before learning (only the feedback controller is applied) falls within the -70 to 70 μm range. As can be seen from Fig. 7, the position error drops to within -4 to 4 μm during the tenth run. If learning is continued thereafter, the position error hardly reduces further, due to noise and restrictions in measurement resolution (0.5 μm).

A remarkable fact is that, although there is a large difference in tracking error, the currents that are fed to the motor in run one and in run ten differ only a little. However, a principal difference

is that, in run one, these currents are determined completely by the feedback controller, whereas, in run ten, they are determined almost completely by the feed forward controller

2) High Velocities: At high speed and, accordingly, large accelerations, the controller has to deal with other effects in the system compared to low-speed motions. As mentioned in the previous section, we had to deal with Stribeck curve effects and cogging in low speed applications. At high speed and large accelerations, the learning controller has to deal mainly with viscous friction and acceleration forces compensation. The disturbance resulting from cogging is relatively small at high speed. After all, cogging disturbance appears at higher frequencies in this case, compared to low-velocity movements. Because of the larger disturbance suppression of the plant for higher frequencies, the amplitude of the cogging disturbance is reduced considerably. This can be illustrated by considering the transfer function that describes the ripple disturbance suppression. For the idealized second-order system controlled by the PID controller, this transfer is given by:

$$\frac{E}{F_{\text{ripple}}} = \frac{-s}{ms^3 + K_d s^2 + K_p s + K_i} \qquad (6)$$

With the PID feedback controller tuned as discussed in Section III-A and a mass m = 37 kg, the Bode diagram of Fig. 8 results. At a low speed of 0.2 m/s and a spatial cogging frequency of 16 mm, the ripple disturbance as seen in the position error signal has a frequency of 12.5 Hz (79 rad/s). At a high speed of 1.0 m/s the cogging disturbance in the position error signal has a frequency of 62.5 Hz (393 rad/s). So, the amplitude of the position error due to cogging is smaller at higher velocities. From stability analysis [14], it becomes clear that the learning feedforward controller may cause system instability (regardless how small the learning rate is chosen) for learning of high frequency components. So, learning of cogging disturbances might result in system instability for high-speed applications. Experimental results showed that instability indeed occurred. Fig. 9 depicts results of a learning process leading to instability. A movement back and forth over 0.5 m in 1.65 s, in which velocities of almost 1 m/s and accelerations of approximately 10 m/s^2 are reached (see the Appendix), is covered repeatedly. The experiment was stopped after the twentieth run, in order to not damage the motor. However, the results indicate that instability occurs *after* the cogging disturbance component had been largely learned.

These figures make clear that learning of high-frequency components, which is only possible when a large number of splines are used, is problematic. Two solutions can be pursued in this case:

- *Reduce the number of splines* used on the setpoint position input space [14] and, therefore, drop the demand for learning the cogging disturbance. As mentioned before, at high speed, the cogging disturbance is relatively small and, therefore, this can be justified.

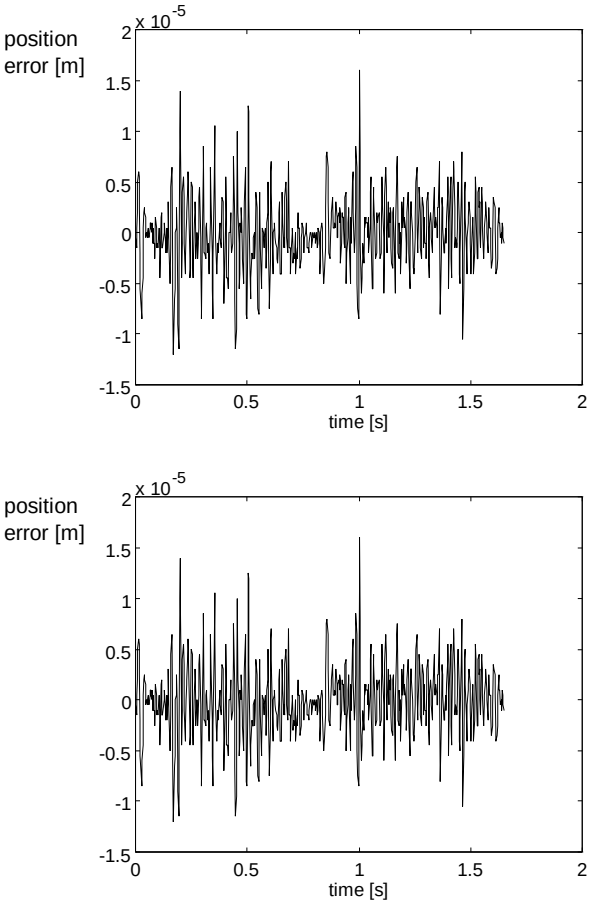

Fig. 10. Learning process when applying a reduced number of splines: position error during (upper) tenth and (lower) one-hundredth run.

- Learn the cogging disturbance as well as possible, and *use a criterion* to stop learning before instability occurs.

The latter option is rejected because the learning controller would then be unable to learn slowly time-varying processes anymore after the criterion determines to stop learning.

The first option is considered, i.e., drop the demand for accurately learning the cogging disturbance and reduce the number of splines on the setpoint position input space. Tuning of the network eventually resulted in spline definitions comparable to the original network, but with only 50 splines on position input space (whereas the original network used 300 splines here). Fig. 10 shows the resulting position error; learning already resulted in optimal behavior after ten learning runs (position error fluctuates within –15 to 15μm) and instability, indeed, did not occur anymore.

V. Conclusions

In this paper, we demonstrated the usefulness of an on-line learning controller for motion control of a linear motor. The learning controller is able to improve system performance drastically, especially in the case of repetitive low-speed motions. The position error reduces by a factor of approx-imately 25. Therefore, without any constructional measures and with little modeling and no identification, a surprisingly good rejection of (reproducible, slowly varying) disturbances, such as cogging and friction, can be achieved.

The consequences of using a learning feedforward structure is that the feedback component of the controller can be designed

with emphasis on robustness, and the learning feedforward component can be used to achieve performance. However, the structure of the learning controller is such that stability problems may arise with high-velocity movements. Decreasing the number of basis functions stabilizes the learning system, but implies some loss of performance as well.

The learning feedforward component consists of a *B*–spline network with second-order basis functions. The network that was used during simulations and experiments was obtained by tuning. Therefore, it may not be the optimal network for the linear motor setup. Further work will be directed towards more systematic design methods for learning feedforward control.

The results generally indicate that an on-line learning controller based on neural network concepts is useful for repetitive tracking control tasks.

APPENDIX

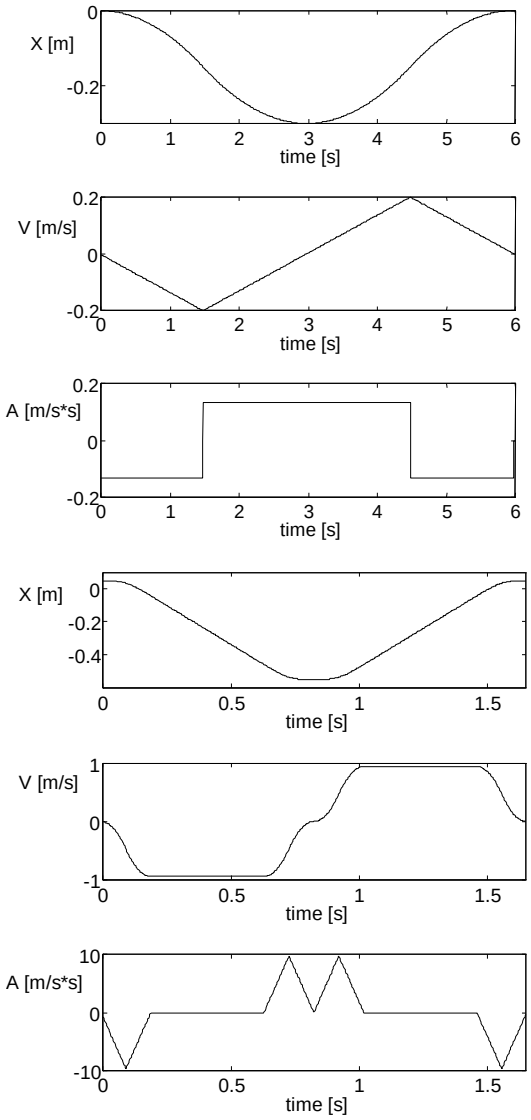

Fig. 11. Setpoint profiles for position, velocity, and acceleration during experiments: (upper) low speed and (lower) high speed.

REFERENCES

[1] M.G.M. Lammers, "Linear lead in ultrasmooth motion", *Mach. Des.*, pp. 60-64, 1994.

[2] D.S. Reay, M. Mirkazemi-Moud, T.C. Green and B.W. Williams, "Switched Reluctance Motor Control Via Fuzzy Adaptive Systems", *IEEE Contr. Syst. Mag*., vol. 15, pp. 8–15, June 1995.

[3] K. Ohnishi, M. Shibata and T. Murakami, "Motion Control for Advanced Mechatronics", *IEEE/ASME Trans. Mechatronics*, vol. 1, pp. 56–67, Mar. 1996.

[4] J.G. Starrenburg, W.T.C. van Luenen, W. Oelen and J. van Amerongen, "Learning feedforward controller for a mobile robot vehicle", *Contr. Eng. Practice*, vol. 4, no. 9, pp. 1221-1230, 1996.

[5] S.A. Nasar and I. Boldea, *Linear Electric Motors: Theory, Design and Practical Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1987.

[6] A. Basak, *Permanent-Magnet DC Linear Motors*. Oxford, UK: Clarendon, 1996.

[7] H. Le-Huy, R. Perret and R. Feuillet, "Minimization of Torque Ripple in Brushless DC Motor Drives", *IEEE Trans. Ind. Applicat*., Vol. IA-22, July/Aug. 1986.

[8] J.Y. Hung and Z. Ding, "Design of currents to reduce torque ripple in brushless permanent magnet motors", *Proc.Inst. Elect. Eng.*, vol. 140, pt. B, no. 4, pp. 260–266, July 1993.

[9] P. van den Braembussche, J. Swevers, H. van Brussel and P. Vanherck, "Accurate tracking control of linear synchronous motor machine tool axes", *Mechatron.*, vol. 6, no. 5, pp. 507–521, 1996.

[10] B. Armstrong–Hélouvry, P. Dupont and C. Canudas de Wit, "A survey of Models, Analysis Tools and Compensation Methods for the control of Machines with Friction", *Automatica*, vol. 30, no. 7, pp. 1083-1138, 1994.

[11] D.E. Rumelhart, G.E. Hinton and R.J. Williams, "Learning representations by back propagating errors," *Nature*, vol. 323, no. 9, Oct. 1986.

[12] T. Poggio and F. Girosi, *A theory of networks for approximation and learning*, Cambridge, MA: MIT, 1989.

[13] M. Brown and C. Harris, *Neurofuzzy adaptive modelling and control*. London, U.K.: Prentice-Hall, 1994.

[14] W.J.R. Velthuis, T.J.A. de Vries and J. van Amerongen, "Learning feed forward control of a flexible beam," in *Proc. IEEE ISIC '96*, Dearborn, MI, 1996, pp. 103–108.

[15] *20-sim Reference Manual*, Controllab Products Inc., Enschede, Netherlands, 1996.

[16] G. Otten, "Performance improvement of mechatronic servosystems by learning feedforward control", Control Lab., Univ. Twente, Enschede, The Netherlands, Rep. BSC 014R96, 1996.

Gerco Otten was born in 1971. He received the M.Sc. degree in mechanical engineering in 1994 from the University of Twente, Enschede, The Netherlands, where he completed a two-year post-graduate course on mechatronic design in 1996.

In 1996, he joined Hollandse Signaalapparaten, Hengelo, The Netherlands, where he is a Design Engineer.

Theo J.A. de Vries (M'96) was born in Wolvega, The Netherlands in 1966. He received the M.Sc. and Ph.D. degrees in electrical engineering from the University of Twente, Enschede, The Netherlands, in 1990 and 1994 respectively, following a special program that combined courses of the Faculties of Electrical Engineering and Mechanical Engineering.

Since 1994, he has been an Assistant Professor in Intelligent Control and Mechatronics in the Control Laboratory, University of Twente. His main research interest is the development of controlled electro-mechanical systems, in particular using learning controllers.

Job van Amerongen (M'90) received the Master's and Ph.D. degrees in electrical engineering from Delft University of Technology, Delft, The Netherlands, in 1971 and 1982, respectively.

From 1971 to 1973 he served as an Officer in the Royal Netherlands Navy. From 1973 to 1987 he was Assistant Professor, later becoming an Associate Professor, in the Control Laboratory, Department of Electrical Engineering, Delft University of Technology, where he worked on applications of modern control theory, especially model reference adaptive control, in ship control systems and electrical power production

systems. Since 1987, he has been a Professor in Control Engineering in the Department of Electrical Engineering, University of Twente, Enschede, The Netherlands. His current research interests are applications of modern control theory, especially intelligent control, in mechatronic systems. As Head of the Control Laboratory, he is also involved in research in modeling and simulation of dynamical systems and in real-time parallel computing. He is also director of the Mechatronics Research Centre Twente, a joint venture of four departments at the University of Twente, where he is also Dean of the Department of Electrical Engineering. He is the author and coauthor of several papers on adaptive and intelligent control systems and automatic steering of ships and the coauthor of a book on adaptive control systems. He is also the author of three courses on systems and control offered by the Dutch Open University.

Dr. van Amerongen is a member of the Royal Institute of Engineers, The Netherlands.

Adrian M. Rankers was born in Bad Gandersheim, Germany, in 1960. He received the M.Sc. degree with honors from the Delft University of Technology, Netherlands, in 1985 and the Ph.D. degree from the University of Twente, Enschede, The Netherlands, in 1997, both in mechanical engineering.

Following receipt of the M.Sc. degree, he joined the Centre for Manufacturing Technology, Philips Electronics N.V., Eindhoven, The Netherlands. He is currently a group leader within the Department Mechatronics, which supports the development of high precision positioning devices, such as compact disc modules, wafersteppers, and component mounters. The special focus of his group is machine dynamics and control systems and the interaction between these disciplines.

Erik W. Gaal (S'85–M'85) was born in Vlissingen, The Netherlands, in 1959. He received the M.Sc. degree in electrical engineering from Eindhoven University of Technology, Eindhoven, The Netherlands, in 1985.

From 1985 to 1991, he was with Philips Research, Eindhoven, The Netherlands, where he worked in the fields of coded modulation, error-correcting codes, and modulation codes for application in optical recording systems. In 1991 he joined the Department of Mechatronics, Philips Centre for Manufacturing Technology, Eindhoven, The Netherlands, where he is involved in the design and implementation of control systems for motion applications, such as component mounters and wafer steppers/scanners.