Proceedings of the 1992 IEEE
International Conference on Robotics and Automation
Nice, France - May 1992

Iterative Self-Improvement of Force Feedback Control in Contour Tracking

Friedrich Lange, Gerhard Hirzinger
Institute of Flight Systems Dynamics
DLR (German Aerospace Research Establishment)
W 8031 Oberpfaffenhofen, Germany

Abstract

A very general 3 level learning method is presented, aiming at self-improvement of the parameters of a force feedback controller demonstrated in contour tracking tasks. It is assumed that no model is known a priori, neither of the robot nor of the contour to be tracked. The system identifies such a model, including information about its reliability. Model and estimated noise are used to generate optimal control actions for the sample trajectory. They are then used for estimation of the parameters of the controller. This controller then produces a new trajectory, which in turn may be optimized and trained thereafter. Kalman filter techniques are applied in all adaptation levels involved. Learning is possible off-line or on-line. Model and controller may be based on linear difference equations or include nonlinear mappings as associative or tabular memories or neural networks. It is shown, that even for a linear controller substantial improvements can be attained, the more as no assumptions are needed about the bandwidth.

1. Introduction

Conventional robot controllers in general show up fairly robust performance when moving the robot to desired positions. If force-feedback via the position control interface and corresponding compliance concepts is realized, performance especially in terms of bandwidth tends to decrease considerably, partly due to excessive delays in the internal control loops. A very fast yet stable controller might request an inverse model of the dynamics of the robot, and, even better, direct joint torque control [1]. This assumes that a precise model is known, and that it can be calculated within the sampling time. If both is true, the method presented in this paper might be inferior. For many applications, however, no such model is available because of limited calculation power or unknown dynamics.

This is the justification of the presented learning system. On one hand it does not need an a priori known model, on the other hand the controller generated is as simple as possible.

Iterative improvement of controllers is not new. There exist many learning systems to refine the tracking of known trajectories by *feedforward* control (trajectory learning), as e. g. Arimoto et al. [2], Atkeson et al. [3] or others.

For improving *feedback* control, in general a coarse model is used to determine the corrections of the controller. This model is assumed to be known or to be identifiable in advance (see e. g. [12]). The speed of adaptation of the controller is dependent on the supposed accuracy of the model. If the model is difficult to identify, no improvement is possible. Otherwise, uncertain model parameters can lead to instabilities.

For adaptive feedback control, there is another common approach for uncertain models. The dynamics of the controller can be reduced by a given reference model (see e. g. [7]). This, however, may reduce the bandwidth of the whole loop.

What are the advantages of the method proposed here? A model built up during the learning process is used, weighted with its reliability. This yields robustness against noise in the data as well as poor excitation during identification of the process or inadequate model assumptions. Of course, if the data are not sufficient, the improvement will be minimal. In any case control will be stable.

Beyond that, no explicit design of a controller is required, and no formulation of a sub-goal for minimization of the trajectory errors is needed (see e. g. Ersü and Tolle [4] or the time inversion training [11]). Instead learning of the controller by reduction of control errors takes into account all preceding control actions as well as the estimated noise, which is an implicit design criterion.

For the reason of clarity, model and controller are chosen to be linear here. The extension of the learning system to nonlinear models, using associative or

0-8186-2720-4/92 $3.00 ©1992 IEEE

tabular memories or neural networks is possible and has been demonstrated for a different process [9].

2. The task to be solved

In our lab a 6 axis industrial robot (manutec r2) is position controllable with sampling intervals of at least 8 ms, in which cartesian position commands are sent to the internal cascaded control system (so called cartesian interpolation system). As typical for many classical robot controllers there is a delay time of several sampling periods between the commands and the beginning of execution.

Forces and torques are measured with a 6-dof compliant sensor. For the endeffector system used, 1 mm of translation corresponds to about 2 N of stationary force.

For simplicity movements are restricted to the x-y-plane. According to Mason's constraint frame concept [10] further distinction is made between force and position control. Neglecting friction the direction of the measured force vector defines the time-variant one-dimensional frame of force control (via positional commands), while the remaining one-dimensional frame is purely position controlled. This reduction to a minimal number of controlled variables was chosen mainly for on-line learning of the control laws.

The force controller is defined to be linear,

$$\begin{aligned} u(k) &= r_{e1} \cdot e(k) + r_{e2} \cdot e(k-1) + \cdots \\ &\quad + r_{u1} \cdot u(k-1) + r_{u2} \cdot u(k-2) + \cdots , \end{aligned} \tag{1}$$

with $e(k)$ denoting the difference between the actual force at time instant k and the desired value (10 N here), while $u(k)$ is the controller's output.

For position control the desired position is simply incremented along the estimated tangential of the contour. The total set-up is shown in Figure 1 showing the robot at the starting position. The experiments shown later are reduced to tracking of the curved parts at the lower part of the figure. For the chosen tangential speed of 0.3 mm/step, corresponding to 37.5 mm/s, this part of the contour has curvatures up to 5°/step, corresponding to more than 600°/s.

3. The learning system

The determination of the parameters of the linear controller (1) is performed by the learning system SLC (= supervised learning control), which is recommended because the availability of a correct model is not assured, as well as the influence of the curvature of the contour is not known.

The learning system consists of three levels:

1. determination of the controller
2. determination of actuator signals
3. determination of a process model

Each level uses information provided by the uppernext one as it is shown in Figure 2.

The third level estimates a model of the process, using the i/o-data of the preceding attempt to control the plant. The model is defined by the parameters $\hat{b}_i$ and $\hat{a}_i$ of the difference equation

$$\begin{aligned} y(k+1) &= \hat{b}_0 + \hat{b}_1 \cdot u(k-n_t) + \hat{b}_2 \cdot u(k-1-n_t) + \cdots \\ &\quad + \hat{a}_1 \cdot y(k) + \hat{a}_2 \cdot y(k-1) + \cdots , \end{aligned} \tag{2}$$

where n_t is the time-delay, caused by signal processing and other (e. g. kinematic) computations in the loop. Here, y denotes the one-dimensional process output,

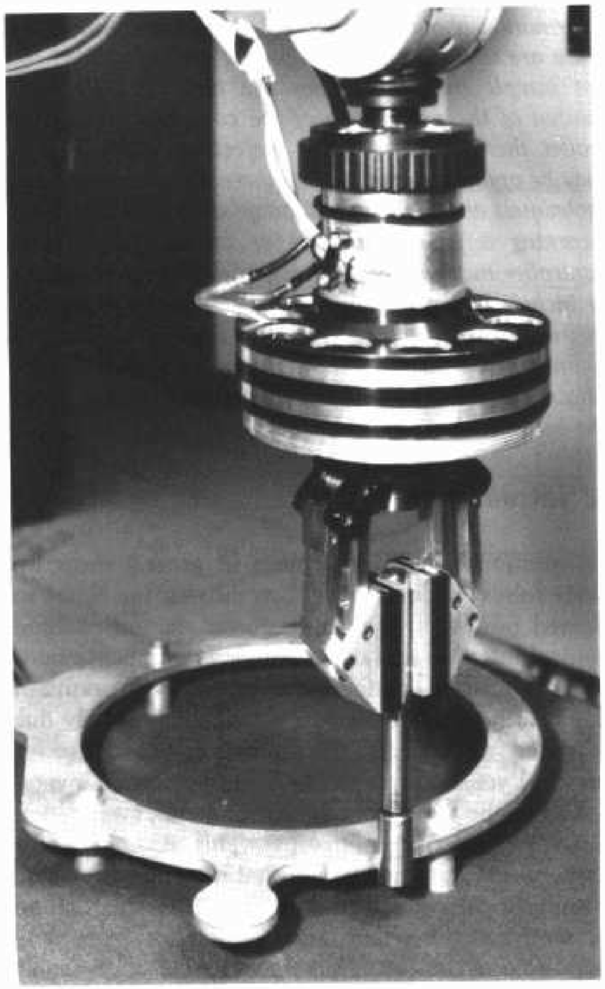

Figure 1. Robot endeffector and contour to be tracked

the absolute value of the force vector, while u denotes the one-dimensional controller output.

Experimental investigations with different industrial robots in our lab indeed have shown that for a restricted workspace (i. e. similar joint configurations) second or third order linearized model approximations are adequate.

This information about the process can be used to optimize the trajectory, from which the model was estimated. To do so, the model is converted to the parameters $\hat{g}_i$ of the impulse response function. This yields a system of linear equations

$$\begin{bmatrix} e(1+n_t) \\ e(2+n_t) \\ \cdots \\ e(n+n_t) \end{bmatrix} = \begin{bmatrix} \hat{g}_1 & & & \\ \hat{g}_2 & \hat{g}_1 & & \\ \cdots & & \cdots & \\ \hat{g}_n & \cdots & \cdots & \hat{g}_1 \end{bmatrix} \cdot \begin{bmatrix} u_{opt}(0) - u(0) \\ u_{opt}(1) - u(1) \\ \cdots \\ u_{opt}(n-1) - u(n-1) \end{bmatrix}, \tag{3}$$

where $e(k)$ denotes the control error, and $u_{opt}(k)$ are the unknowns, determining the optimal control actions for eliminating the disturbances of the preceding control.

For asymptotically stable processes, which for constant control actions converge to a constant control error, many of the elements of the impulse response matrix can be dropped, leading to a band matrix. Hence the increase of the computational effort is only linear with respect to the number of control cycles.

Finally, the controller can be trained from the examples, provided by the second level. This again is a problem of parameter estimation which can be performed by standard estimation techniques as the correction of the model. Thereafter the controller should be able to issue the optimal control actions, when the same disturbances as in the preceding trajectory occur. This will not yield vanishing errors due to the design as a feedback controller. Estimating the control parameters using a stochastic approach results in a linear controller (1) which may be applied to arbitrary non-trained contours, too.

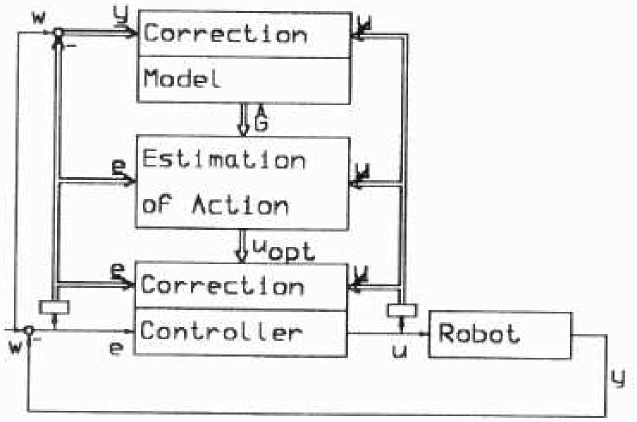

Figure 2. Structure of the learning system (upper part: off-line learning, lower part: on-line control)

For all estimations Kalman filter technology was chosen, minimizing the square error of the corresponding parameters. In this way model and controller are estimated in the third and first level. The second level uses the inverse Kalman filter to take advantage of the special from of the impulse response matrix to estimate the optimal control actions.

Treating the calculations as estimation problems allows to deal with noisy data or incomplete models. In each level, corrections take place weighted by the reliability of the input data. As additional output the precision of the determined values can be estimated. The reliability of the input data of one level can be expressed by the output errors of the uppernext level. For the third level itself, the disturbances are estimated by a special algorithm. This yields convergence of the learning system even under unknown conditions.

As an example for the use of the reliability, the correction of the controller is discussed: The variance σ^2 of the disturbance, being an input for the Kalman filter, is determined by the variance of the optimal control actions provided by the second level and by the variance of the disturbances of the control errors, used in (1). The disturbances of the control errors consist of two parts i. e. uncorrelated and correlated noise, both being estimated in the third level by comparing predicted and real measurements .

The Kalman filter algorithm

$$\begin{aligned} \underline{\theta}(k+1) = \underline{\theta}(k) + P(k) \cdot \underline{\psi}(k+1) \\ \cdot (\underline{\psi}^T(k+1) \cdot P(k) \cdot \underline{\psi}(k+1) + \sigma^2)^{-1} \\ \cdot (\mu(k+1) - \underline{\psi}^T(k+1) \cdot \underline{\theta}(k)) \end{aligned} \tag{4}$$

$$\begin{aligned} P(k+1) = P(k) - P(k) \cdot \underline{\psi}(k+1) \\ \cdot (\underline{\psi}^T(k+1) \cdot P(k) \cdot \underline{\psi}(k+1) + \sigma^2)^{-1} \\ \cdot \underline{\psi}^T(k+1) \cdot P(k) \end{aligned} \tag{5}$$

automatically adapts the parameters in relation to the assumed disturbance of the input data, the corrections being minimal for high disturbance σ^2. (For the estimation of the controller $\underline{\theta}$ denotes the parameter vector $\underline{\theta} = (r_{e1}, r_{e2}, \ldots r_{u1}, \ldots)^T$ being estimated, while $\underline{\psi}(k) = (e(k), e(k-1), \ldots, u(k-1), \ldots)^T$ and $\mu(k) =$

$u(k)$ are the coefficients and the left side in (1), respectively, which can be measured.).

4. Structure of control

The task as well as the learning system are explained so far as a one dimensional control problem. However, an interface is necessary, because of the time-variant definition of the force controlled frame.

This interface has to simulate an asymptotically stable process for the learning system to allow the simplification of the second level. Therefore an integrating system is unfavourable. This means that the controller should command fictive (i. e. one-dimensional) positions instead of increments. On the other hand, sensor signals only describe differences between desired and actual position. The resulting control structure is shown in Figure 3. It shows the commanded position vector $\underline{u}$ as the sum of increments, which are provided by differentiating the scalar controller output u. The force vector $\underline{f}$ is converted into the absolute value $y = |\underline{f}|$ and the unit direction vector $\underline{r} = \underline{f} / |\underline{f}|$.

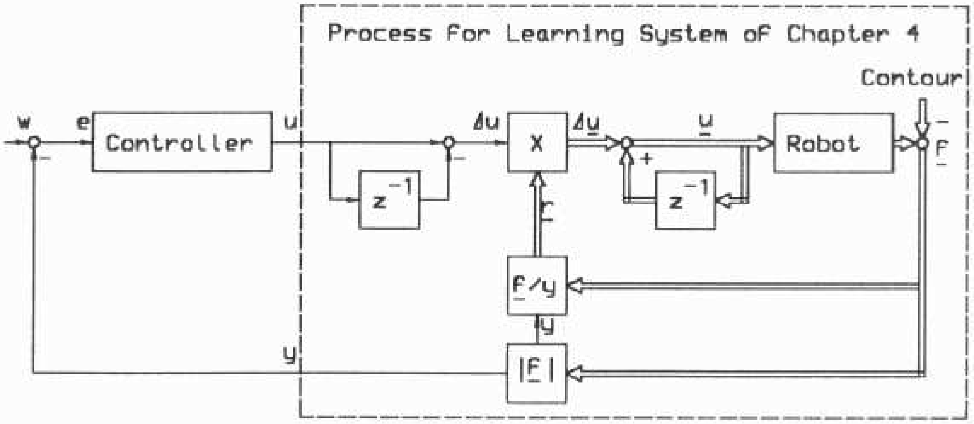

Figure 3. Structure of Force Control (z^{-1} denoting a delay of one sample period)

5. Experimental results

The task was presented in chapter 2.. It is solved by

1. Select arbitrary controller parameters,
2. Control with this controller, until the absolute value of the force leaves the allowed range, or the end of the contour is reached,
3. Measure the performance,
4. Go back near the starting position,
5. Start the learning algorithm to get a new controller,
6. Continue at point 2 as long as the performance is not satisfying. Otherwise store the parameters.

It should be emphasized that the starting positions are varied intentionally for every iteration, so that no feedforward control is possible. This yields different parts of the contour to be tracked and, finally, a controller to be estimated which is able to track arbitrary contours within the same bandwith.

For every trajectory the root mean square error is computed. It stands for the mean force error in N. Because of the similar sequence of curvatures in the training phase, these errors can be compared. During the learning phase, they decrease considerably , as can be seen in Table 1.

The first iterations are plotted in Figure 4. It can be seen that the error is reduced substantially within the two first corrections. Then the controller is able to track the whole contour. The remaining control error can hardly be reduced with this simplified

	Iteration	Mean force error
off-line	1	4.380
	2	3.337
	3	1.167
	4	0.698
on-line	1	1.545
	2	1.058

Table 1. Mean force error in N during learning of the controller

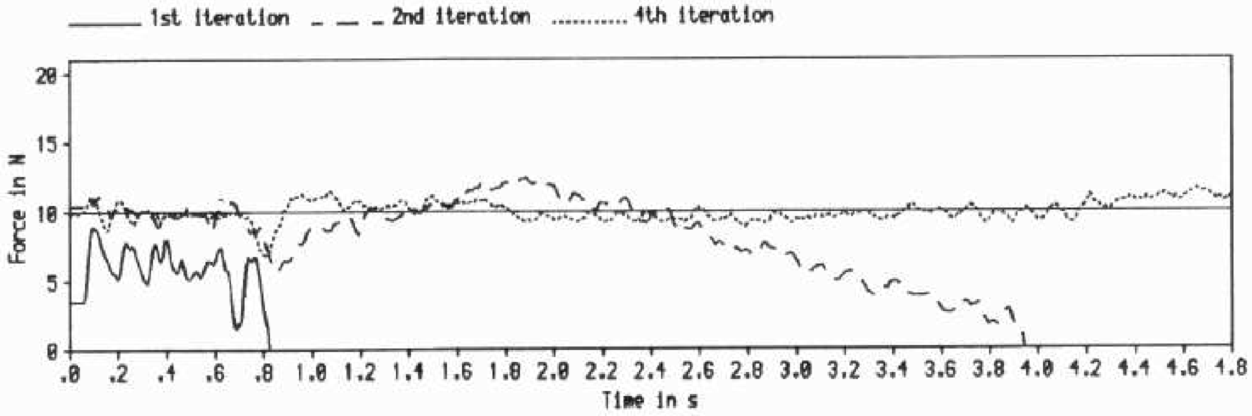

Figure 4. Absolute values of the force vector during off-line learning (experimental data)

approach due to noise or time delay. It may be interpreted, however. After about 0.8 s the robot tool passes the edge in the contour, causing an immeadiate decrease of force. Thereafter the long concave curvature effects a small overshoot of force. In contrast, the following convex area is the reason for the forces being too small. Finally the tracking experiment ends soon after the next concave region.

This is one of several experiments. Others may be better, tracking the whole contour already in the 2nd attempt, or worse, needing 4 or 5 iterations to learn. Nevertheless the final quality is always as good as in the dotted curve in Figure 4.

In the first iteration, the behaviour depends only on the initially chosen controller, as learning so far takes place only off-line. So the controller looses contact at the edge (after about 0.8 s in Figure 4). This can be improved by learning on-line.

The lower part of Table 1 and Figure 5 shows the first iteration, when learning begins as soon as there is enough information. For the model to be identified, typically 15 sampling intervals are needed until the controller can be modified the first time.

For on-line learning on the computer used (microVAX III) the sampling time has to be changed to 16 ms, thus increasing the tangential and the angular motion per step to 0.6 mm and 10°, respectively. This has to be kept in mind when comparing the performances of Figure 4 and Figure 5.

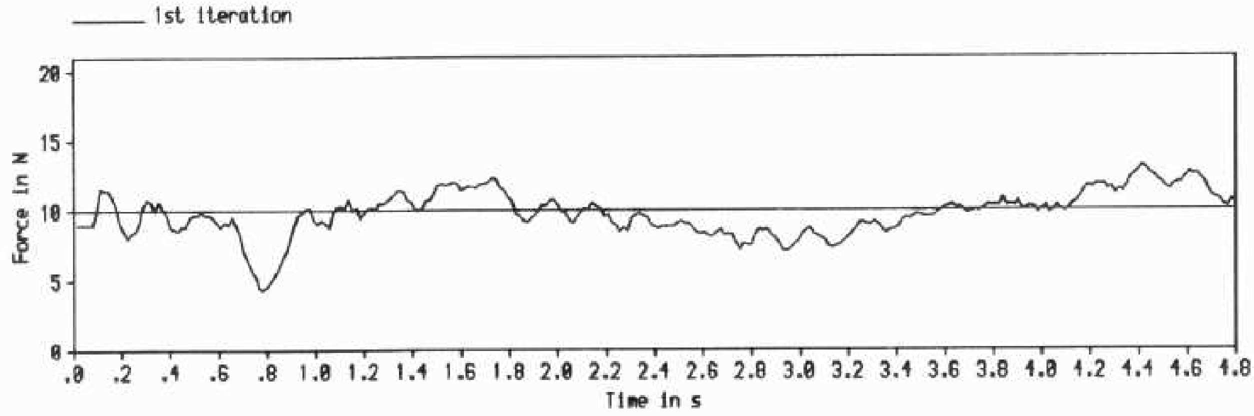

Figure 5. Absolute values of the force vector during on-line learning (starting with the same unsatisfactory initial controller as in Figure 4 and using a less suitable sampling time)

6. Discussion

The learning system as proposed here yields substantial improvements in relation to the initial chosen controller, because no a priori information concerning the reachable model accuracy, noise, or curvature of the contour is needed.

The learning system produces all necessary informations for the correction of the controller, as far as this is possible due to noise. So our approach does not need a given model but can be superior than methods with process independent adaption laws.

For higher speed, the initial controller has to be rather good, for being able to track at least for several steps. In this case it may be useful to learn a controller for low speed first, and to use this controller as the initial one for the higher speed. Likewise speed can be increased several times, until the speed, as is desired for the application, is reached.

The main performance limit lies in the sampling period and the delay between commands and reactions. For higher sampling rates and higher speed along the contour, however, two other problems will occur.

First, the dynamics of the robot strictly speaking are nonlinear, so that the linear approach will not be satisfactory any more. The concept of our learning method, however, is capable of handling nonlinear models and controllers as associative or tabular memories or neural networks. Associative mappings have been successfully used in the past [8]. Presently, application of neural nets is investigated, final results not being available yet.

It is not necessary, however, to process all joint values in the model, as would be necessary for exact decoupling. Typically only one or two nonlinear influences will be sufficient, e. g. the direction of control movements or the absolute value of the force. The first takes into account that different directions are controlled with different joints, which have different characteristics, the latter considers changes of the dynamics due to the closed kinematic chain when the robot is in force contact.

The other problem is the one-dimensional formulation of force control. The used approach of separation between force and position control is the best solution for one-dimensional handling of the two controllers. For fast movements along curved surfaces it is not adequate. In this case at least the frame for force control should be time-invariant.

Although the learning scheme presented has been discussed here in the context of force controlled contour tracking, it may be applied in any kind of sensory feedback. The space robot technology experiment ROTEX to fly in early 93 will use these techniques in the telerobotic groundstation for self-improvement of the local on-board sensory feedback loops [5][6].

7. Bibliography

[1] An, C. H., C. G. Atkeson, and J. M. Hollerbach *Model-Based Control of a Robot Manipulator* The MIT Press, 1988

[2] Arimito, S., T. Naniwa, and H. Suzuki *Selective Learning with a Forgetting Factor for Robotic Motion Control* IEEE Int. Conf. on Robotics and Automation, Sacramento, California, April 1991

[3] Atkeson, C. G., E. W. Aboaf, J. McIntyre, and D. J. Reinkensmeyer *Model-Based Robot Learning* 4th Int. Symposium of Robotic Research, Santa Cruz, Aug. 1987

[4] Ersü, E. and H. Tolle *A New Concept for Learning Control Inspired by Brain Theory* IFAC-Congress, Budapest, July 1988

[5] Hirzinger, G., J. Heindl, and K. Landzettel *Predictive and Knowledge-Based Telerobotic Control Concepts* IEEE Int. Conf. on Robotics and Automation, Scottsdale, Arizona, May 1989

[6] Hirzinger, G., G. Grunwald, B. Brunner, and J. Heindl *A Sensor-Based Telerobotic System for the Space Robot Experiment ROTEX* Second Int. Symp. on Experimental Robotics, Toulouse, France, June 1991

[7] Isermann, R. *Parameter Adaptive Control Algorithms - A Tutorial* Automatica, Vol. 18, 1982

[8] Lange, F. *A Learning Concept for Improving Robot Force Control* IFAC Symposium on Robot Control, Karlsruhe, Oct. 1988

[9] Lange, F. *A Learning Control Concept with Tabular Knowledge Base* VDI Berichte Nr. 897, 1991 (in German)

[10] Mason, M. T. *Compliance and Force Control for Computer Controlled Manipulators* IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-11, 1981

[11] Miller III, W. T. *Sensor-Based Control of Robotic Manipulators using a General Learning Algorithm* IEEE Journal of Robotics and Automation, Vol. RA-3, No.2, 1987

[12] Raibert, M. H. *A Model for Sensorimotor Control and Learning* Biological Cybernetics, Vol.29, 1978