Design of a Predictive Hierarchical Controller Using FEMLAB

B. Rehák

A hierarchical controller with two levels is proposed. One level is based on dynamic optimization while the second is responsible for tracking the optimal trajectory and rejecting disturbances. Its implementation using the FEMLAB system is described. Some simulations are presented at the end of the paper, together with an evaluation of the performance.

Keywords: hierarchical control, boundary-value problem, dynamical optimization, FEMLAB.

1 Introduction

The theory of dynamic optimization problems has been developed quite well. However, it is still not much used in applications. There are several reasons for this – the first is that solving a general dynamic optimization problem requires solving a general boundary problem for a system of ordinary differential equations. This problem is difficult, due to the need for proper discretization and also to the huge time consumption. Another drawback is that the control law cannot be formulated in the feedback form. That is why this theory was almost abandoned, despite its successful applications to the optimization of, e.g., space flights in the 1960's. This theory with some applications to this area has been introduced in many publications, e.g., [2].

The boundary problem arising from the dynamic optimization can be solved using the finite element method (FEM) similarly as it is used for solving partial differential equations. Hence it is possible to use PDE solving tools for dynamic optimization. Thanks to the vast efforts made in computer equipment in the last twenty years, powerful and user-friendly software packages for solving boundary problems for ordinary differential equations have appeared. One such software system is Femlab. Its close relationship to the well-known Matlab system makes its application in complex control problems straightforward.

Even if the modern computer equipment is used for solving the boundary problem, computations can take too long to be used for directly designing a control law. Hence we have designed a hierarchical controller. The theory of hierarchical control systems was developed in the early 1980's in the works of Singh, Siljak, Findeisen and others. A good summary of this theory can be found in [4].

In this article we design a control algorithm based on general dynamic optimization in the upper level. The lower level was designed via the theory of LQ control. We use the FEMLAB 2.3 package for solving the optimization problem, and the implementation is also described. Simulations showing the advantages of the hierarchical control approach are mentioned. This approach can handle nonquadratic cost functionals, e.g. functionals containing a barrier function. This paper summarizes and extends the results from [6].

2 Hierarchical control law

The real-time optimal control problem was divided into two subproblems. The first problem is to compute the optimal trajectory prediction. This involves dynamic optimization. The Femlab system was used to carry out computations at this level. The second subproblem is to track this optimal trajectory. This is the task of the LQ controller. These two subproblems are solved in different levels. The first is referred to as the upper while the latter is the lower level of the controller. The tracking in the lower level is independent of the computations of the upper level. Thus both these tasks may run in parallel. The scheme of the hierarchical control system is shown in Fig. 1 [4, 7].

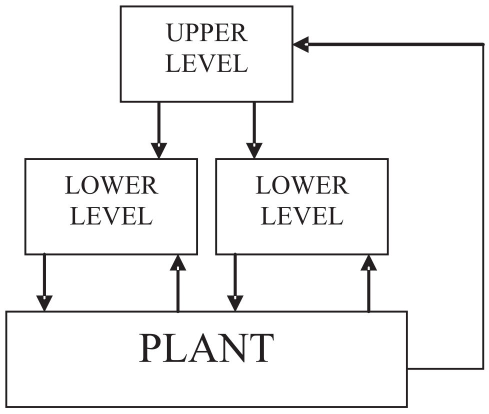


Fig. 1: Scheme of a hierarchical control system

We will now outline the main features of both levels now.

Upper level

- determines the optimal trajectory for a long period ahead,
- time-consuming calculations are carried out here,
- the whole structure of the system is taken into account, including nonlinearities etc.,
- only the slower dynamics are considered.

Lower level

- is responsible for tracking the optimal trajectory,
- compensates the disturbances,
- the control law should be as simple and as fast as possible and should have the feedback form,
- when designing the control law design the most important connections are taken into account while the nonlinearities can be replaced by their linearizations,
- the influence of the fast modes should be also be taken into consideration.

The delay caused by computation of optimal control law in the upper level design should also be taken into account. While this computation is being carried out, the lower level uses the trajectory computed in the previous step. The time during which the trajectory computed in one computational step of the upper level is used is denoted by T_v. This period must be longer than the longest computation in the upper level.

3 Control algorithm design of the upper level

In this paragraph we will introduce the optimal control problem, which makes up a central part of the upper-level design. Then we will focus on how the results are applied in order to complete the upper level design.

Let $T > 0$. We consider the system described by the state equation

$$\dot{x} = f\big(x(t), u(t)\big)$$

together with the initial condition

$$x(T) = x_0 .$$

Our aim is to design a control $u(t)$ such that the cost functional

$$J(T) = \int_T^{T+T_H} j\big(x(t), u(t)\big)\,\mathrm{d}t$$

is minimized. In the definition of the cost functional, T_H stands for the prediction horizon and j denotes a continuously differentiable function such that the integral exists. The horizon T_H must be longer than the longest time necessary for the calculations described below. It can, however, be significantly greater, as mentioned in [5].

The optimization is performed under the condition that the state equation is satisfied for every t between T and T_H. The problem described above will be referred to as the upper-level problem at time T on the interval (T, T_H) with the initial condition $x(T) = x_0$ and denoted by ($\mathbf{P}[\mathbf{T}, \mathbf{x_0}]$).

First, we introduce the solution of the problem ($\mathbf{P}[\mathbf{T}, \mathbf{x_0}]$). Then we will demonstrate how it can be used for the upper-level definition.

Proceeding as in [2], we infer that the problem ($\mathbf{P}[\mathbf{T}, \mathbf{x_0}]$) is equivalent to the problem of unconstrained minimization of the Lagrange functional L that is defined as follows:

$$L(T) = \int_T^{T+T_H} \Big[j\big(x(t), u(t)\big) + \lambda^T(t)\cdot\big(\dot{x}(t) - f\big(x(t), u(t)\big)\big)\Big]\,\mathrm{d}t .$$

The vector-valued function λ (having the same size as the state vector) is the Lagrange multiplier, sometimes also called the co-state or the adjoint state. The functions u, x, λ that solve the minimization problem also satisfy the following set of equations on the interval (T, T_H) ($D_x F$, or $D_u F$ denotes the derivative of function F with respect to variable x or u.):

$$\dot{x} = f\big(x(t), u(t)\big)$$
$$\dot{\lambda} = D_x j\big(x(t), u(t)\big) + D_x f\big(x(t), u(t)\big)\,\lambda(t)$$
$$0 = D_u j\big(x(t), u(t)\big) + D_u f\big(x(t), u(t)\big)\,\lambda(t)$$

with boundary conditions defined as follows:

$$x(T) = x_0$$
$$\lambda(T + T_H) = 0$$

while the value of the state at the end of the interval (T, T_H) and also the value of the co-state at time T are not defined. Then the boundary value problem for this set of equations is correctly defined. See [2, 9] for details.

We will now describe the functionality of the upper level. Since the solution of the optimization problem takes a considerable time, the algorithm for the solution can be started with period T_v, as described above. The period must be longer than the time necessary for carrying out the computations, but it must be also shorter than the prediction horizon T_H. During that period, the following actions take place (let us assume that the current time is kT_v):

1. The prediction of the state variables at time $(k+1)T_v$ is evaluated with the help of the feedforward computed at time $(k-1)T_v$. We denote this prediction by $\xi((k+1)T_v)$.
2. The optimization problem $\mathbf{P}[\mathbf{T}(\boldsymbol{k}+\mathbf{1})\boldsymbol{T_v},\ \xi((\boldsymbol{k}+\mathbf{1})\boldsymbol{T_v}]$ is solved.
3. The optimal feedforward for the lower-level control loops on the interval $[(k+1)T_v, (k+2)T_v]$ is evaluated. We will clarify this step in the following paragraph.

At the time slot $(k+2)T_v$ the optimal feedforward is saved into the buffer and these steps are repeated with new values of the initial and terminal time of the optimization problem. This is a kind of "receding horizon" method. It constitutes an essential part of predictive control theory [5].

4 Design of the lower level

We will now describe the functionality and the design of the lower-level controllers. First we mention that we will make use of the possibility to decompose the system. We briefly mention this decomposition. The most important condition is that the systems must be almost autonomous, and their mutual interaction is only weak. In what follows we assume that the system can be divided into N subsystems. The system matrix (or, if the original system was nonlinear, its linearization) can be written as follows:

$$\boldsymbol{A} = \begin{pmatrix} A^1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & A^N \end{pmatrix} + A^{connection} .$$

The term $A^{connection}$ contains all the interactions that are not taken into account in the lower-level design. We also assume that matrices $\boldsymbol{B}$, $\boldsymbol{Q}$, $\boldsymbol{R}$ can be decomposed similarly, with compatible dimensions so that the i-th subsystem (with neglected interactions) is described by the equation

$$\dot{x}^i = A^i x^i + B^i u^i .$$

Since high speed is the main required feature, we decided to implement this level using time–invariant LQ controllers. LQ controllers also offer the advantage of a straightforward analysis of the cost of the control. See, e.g., [1, 9] for general LQ-control theory and [7, 8] for more results about decentralized LQ control and about constraints imposed by the prescribed structure of the controller.

We assume that the lower lever controller of the i-th subsystem should track the optimal trajectory $x^i_{opt}(t)$ and the optimal control $u^i_{opt}(t)$, which were evaluated in the upper level. The tracking should be such that the cost

$$J^i = \int_T^{T+T_H} \left[\left(x^i(t) - x^i_{opt}(t)\right)^T Q_i \left(x^i(t) - x^i_{opt}(t)\right) + \right. \\ \left. + \left(u^i(t) - u^i_{opt}(t)\right)^T R_i \left(u^i(t) - u^i_{opt}(t)\right) \right] \mathrm{d}t$$

is minimal. Here, symbol $u^i(t)$ denotes the optimal control which is applied to the system, and $x^i(t)$ stands for the real trajectory. We assume that the weighting matrices $\boldsymbol{Q}^i$ and $\boldsymbol{R}^i$ are positively semidefinite and positively definite, respectively.

Using the dynamic programming approach (see e.g. [1]), we can infer that the optimal control is given by the formula

$$u^i(t) = u^i_{feedback}(t) + u^i_{feedforward}(t).$$

The control consists of a feedback term and a feedforward term. For the feedback term it holds:

$$u^i_{feedback}(t) = -(R^i)^{-1} P^i(t) B^{iT} x^i(t)$$

while the feedforward is defined by:

$$u^i_{feedforward}(t) = -(R^i)^{-1} B^{iT} p^i(t) + u^i_{opt}(t)$$

where function $P^i(t)$ is actually the solution of the continuous Riccati equation with the initial condition $P^i(T+T_H) = 0$ for the i-th system. Function $p^i(t)$ solves the following differential equation

$$\dot{p}(t) = (A^i - P^i(t) B^{iT} (R^i)^{-1} B^i)\, p^i(t) + Q^i x^i_{opt} - P^i(t)\, B^i u^i_{opt}(t)$$

with the initial condition $p^i(T+T_H) = 0$. The last term in the equation is nonstandard. It is due to the existence of the desired control, which should also be tracked. Note that this equation is solved in the backward direction.

Strictly speaking, the LQ control on the horizon equal to the optimization horizon in the upper level (denoted by T_H, see below) presented so far should be applied. Nonetheless we assume that the horizon is long enough to replace the time-variant control by the time-invariant control without a significant loss of accuracy. Another advantage is that we obtain a time-invariant gain in the control loop. This feature simplifies the implementation significantly. Hence we replace matrix function $P^i(t)$ by its limit value P^i in the differential equation for function $p^i(t)$. The limit solution P^i solves the algebraic Riccati equation:

$$0 = A^{iT} P^i + P^i A^i - P^i B^{iT} (R^i)^{-1} B^i P^i + Q^i.$$

This trick simplifies the implementation of this level significantly.

5 Analysis of the influence of the decomposition on the total cost

In this section we will analyze the increase in the cost when the control algorithm is designed using a hierarchical approach. We will make some assumptions that will simplify our analysis. First we assume that the total cost is quadratic, i.e., that there exist a symmetric positive semidefinite matrix $\boldsymbol{Q}$ and a symmetric positive definite matrix $\boldsymbol{R}$ such that

$$J(T) = \int_T^{T+T_H} \left[(x_w(t) - x(t))^T Q (x_w(t) - x(t)) + u(t)^T R u(t) \right] \mathrm{d}t$$

where x_w denotes the desired trajectory that is to be tracked by the controlled system. Moreover we assume that the weighting matrices can be decomposed as described above. The dynamic optimization yields an optimal trajectory that should be tracked together with the optimal control that should be applied. This trajectory, or control, is denoted by x_{opt}, u_{opt}. (We will omit the explicit writing of time argument t in what follows.) We then have

$$J(T) = \int_T^{T+T_H} \left[(x_w - x_{opt} + x_{opt} - x)^T Q (x_w - x_{opt} + x_{opt} - x) + \right. \\ \left. + (u - u_{opt} + u_{opt})^T R (u - u_{opt} + u_{opt}) \right] \mathrm{d}t =$$

$$\int_T^{T+T_H} \left[(x_w - x_{opt})^T Q (x_w - x_{opt}) + (x_{opt} - x)^T Q (x_w - x_{opt}) + \right. \\ + (x_w - x_{opt})^T Q (x_{opt} - x) + (x_{opt} - x)^T Q (x_{opt} - x) + \\ + u_{opt}^T R u_{opt} + (u_{opt} - u)^T R u + u^T R (u_{opt} - u) + \\ \left. + (u_{opt} - u)^T R (u_{opt} - u) \right] \mathrm{d}t \le$$

$$\int_T^{T+T_H} \left[2(x_w - x_{opt})^T Q (x_w - x_{opt}) + 2(x_{opt} - x)^T Q (x_{opt} - x) + \right. \\ \left. + 2u_{opt}^T R u_{opt} + 2(u_{opt} - u)^T R (u_{opt} - u) \right] \mathrm{d}t = 2J_{UL} + 2J_{LL}$$

Here, J_{UL} denotes the optimal cost achievable by the optimization in the upper level and J_{LL} stands for the lower-level optimal cost. The following holds for them:

$$J_{UL} = \int_T^{T+T_H} \left[(x_w - x_{opt})^T Q (x_w - x_{opt}) + u_{opt}^T R u_{opt} \right] \mathrm{d}t$$

and

$$J_{LL} = \int_T^{T+T_H} \left[(x_{opt} - x)^T Q (x_{opt} - x) + (u_{opt} - u)^T R (u_{opt} - u) \right] \mathrm{d}t.$$

Nonlinearities would make this reasoning much more complicated. The same holds if the cost is non-quadratic.

6 Upper level implementation

We mention how the control scheme described above can be implemented using the Femlab software package. The implementation is fairly straightforward since this software enables easy cooperation with Matlab. We will describe briefly how the code was generated and what changes are to be made. We used the Femlab 2.1 system (see [3]) as well as its graphical user interface (GUI). Generating the code using the GUI saves a lot of effort.

For simplicity we assume the system to be of order **two**. Implementation of the control algorithm for a higher-order system will then be a straighforward extension of our approach We chose the general form, 6 one-dimensional equations on the interval (0, T_H) in the initial menu. The names of the functions are **x1**, **x2**, **λ1**, **λ2**, **u1**, **u2**. The first two stand for the state, then the pair **λ1**, **λ2** represents the co-

-states, and finally the last two variables are used for control. Then we define constants **xx10** and **xx20**, which will be used for defining the boundary conditions of the state variables.

Next we define the following system of equations, which describes the dynamic optimization problem as introduced above.

There is a slight problem with the definition of the boundary conditions. It is clear from the definition of the state and co-state functions that the following must be satisfied: $\mathbf{x1(0)} = \mathbf{xx10}$, $\mathbf{x2(0)} = \mathbf{xx20}$, $\boldsymbol{\lambda}\mathbf{1(T_H)} = 0$, $\boldsymbol{\lambda}\mathbf{2(T_H)} = 0$. Nevertheless Femlab also requires initial conditions for the states defined at the end of the interval and for the co-states at the beginning. This is due to the fact that the Femlab GUI was optimized for designing programs for solving differential equations of the second order. Thus we have to define the boundary condition as void. We defined there the conditions $\mathbf{x1(T_H)} - \mathbf{x1(T_H)} = 0$, resp. $\boldsymbol{\lambda}\mathbf{1(0)} - \boldsymbol{\lambda}\mathbf{1(0)} = 0$, for the other state and co-state variables by analogy. Then the equations are correctly defined and their definition meets the requirements of the Femlab system. We also have to define the functions that evaluate the desired trajectory.

The following actions were carried out with the code generated by the GUI:

- It was declared as a function with input parameters **time**, **x1time**, **x2time**. The time parameter contains the actual time, and the parameters **x1time**, **x2time** contain the values of the state variables.
- The interval where the boundary problem was solved was changed to (**time**, **time + TH**).
- At the beginning of the function, the value of the state variables is evaluated at the time slot **time + 1**. This is possible since the control on this interval is known from the previous step.
- Then the code generated by the GUI follows. Some modifications were made: the time interval where the problem is solved was changed to **time + 1**, **time + 1 + TH**. The boundary conditions on the state variables are their value at the time **time + 1**, where the constant **TH** contains the length of the prediction horizon.
- Then the feedforward functions were computed and saved.

The Simulink system was used for modeling the system. To simulate both levels properly it would be necessary to run the calculations in two different threads. We did not perform this. The scheme is shown in Fig. 2.

The gains Gain and Gain1 together with the system build up the lower level. The functions firstreference and secondreference select the appropriate feedforward computed in the upper level. The connection from the lower into the upper level is realized by the function MATLAB Fcn, which activates the upper level always after period 1.

7 Simulation results

A simulation example was performed with the system

$$\dot{x}_1 = x_1 + 0.01\, x_2 + u_1$$
$$\dot{x}_2 = 0.01\, x_1 + x_2 + u_2$$

together with the following quadratic cost functional J.

$$J = \int_T^{T+3} \left[\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}^T \begin{pmatrix} q_1 & 0 \\ 0 & q_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}^T \begin{pmatrix} r_1 & 0 \\ 0 & r_2 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \right] \mathrm{d}t.$$

The matrices $\boldsymbol{Q}$, $\boldsymbol{R}$ in the upper-level cost functional were chosen such that $q_1 = q_2 = 10$, $r_1 = r_2 = 0.2$. The lower-level compensators have gains optimal for controlling the system, with the interconnections between these subsystems neglected. The matrices in the quadratic cost functional are chosen as q_1, r_1 resp. q_2, r_2.

We aim to design the control so that the state x_1 will track the reference $x_{1w} = \sin 4t$ and the state x_2 will track the reference $x_{2w} = \cos 5t$.

The equations for the derivatives of the co-states attain the following form

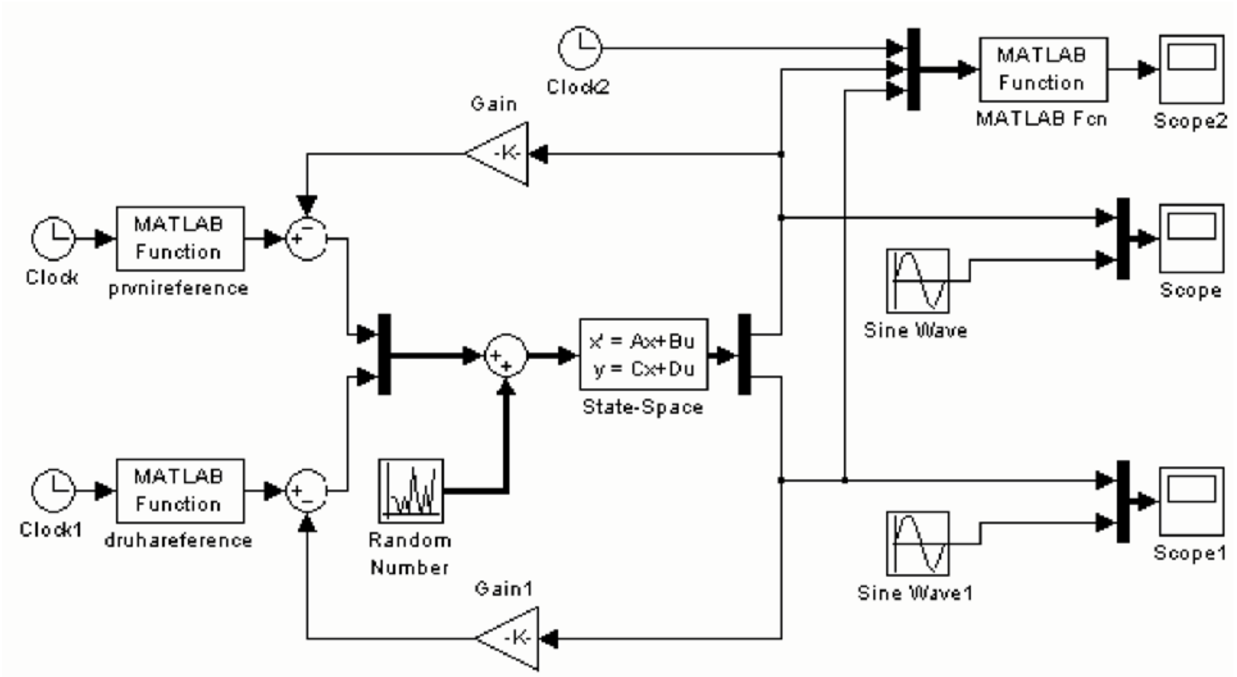


Fig. 2: The simulation scheme

The cost functional can also be augmented by a barrier function. To show how the optimal controller takes this into account we carried out similar simulations of the same system when the term

$$b(x_1, x_2) = \begin{cases} 0.5\left(\dfrac{1}{(x_1 - 0.6)^4 + x_2^4 + 0.05}\right)^4 & \text{if } \dfrac{1}{(x_1 - 0.6)^4 + x_2^4 + 0.05} > 10 \\ 0 & \text{elsewhere} \end{cases}$$

was added to the cost functional. This barrier function penalizes trajectories that are close to the point $x_P = (0.6, 0)$ in the state space. The system as well as the cost functional are very simple, but still this example demonstrates the main features of the proposed approach.

$$\dot{\lambda}_1 = q_1(x_{1opt} - x_{1w}) + \lambda_1 + 0.01\lambda_2 + \frac{\partial b(x_{1opt}, x_{2opt})}{\partial x_1}$$

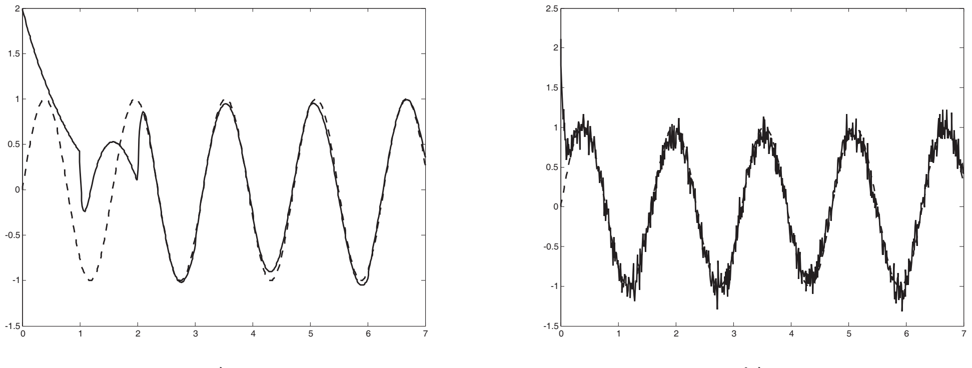
$$\dot{\lambda}_2 = q_2(x_{2opt} - x_{2w}) + 0.01\lambda_1 + \lambda_2 + \frac{\partial b(x_{1opt}, x_{2opt})}{\partial x_2}$$

with zero terminal condition at $T+3$ again.

The state is shown in Fig. 3a without the barrier function and without the presence of the additive noise. (In this and in the following figures, the solid line represents the state x_1 of the system while the dashed line represents the reference.) Fig. 3b shows the state x_1 if the additive noise acts upon the system while the barrier function is not present. The next two figures show the state x_1 in the situation after augmentation of the cost functional by adding the barrier function. The state x_1 is shown in the Fig. 4a without the presence of the additive noise, and in Fig. 4b in the presence of this noise. The state space is shown in Fig. 5a, respectively 5b (with, respectively without, the barrier function activated). Here we can easily see the effect of penalizing the states that are close to the point x_p. The dashed circle joins the points where the barrier function (defined above) attains the value 1.

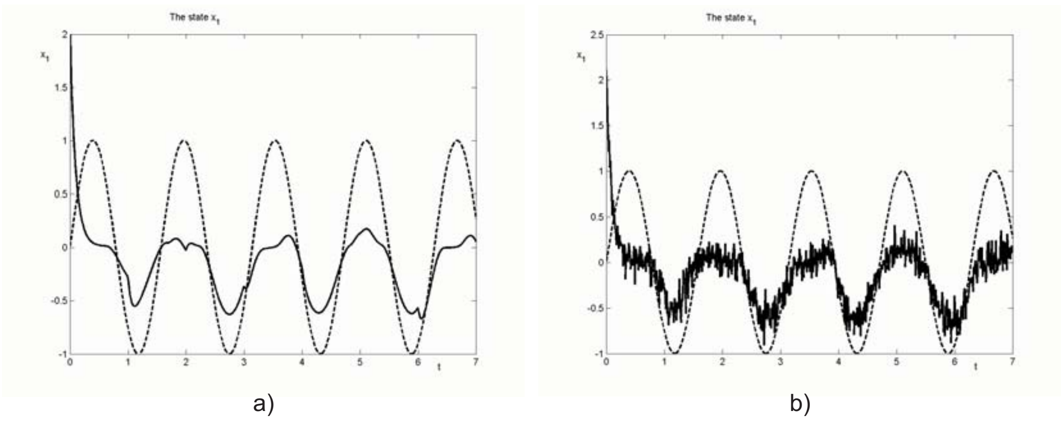
Finally, if the system is controlled by the optimal control computed in the upper level, the state x_1 behaves as depicted in Fig. 6 (solid line), the reference being again represented by the dashed line. The loss of reasonable performance is due to the fact that the system remains virtually uncontrolled during the computations in the upper level while the noise still acts.

The simulations carried out show that the behavior of states x_1 and x_2 is very similar. Therefore the state x_2 is not shown.



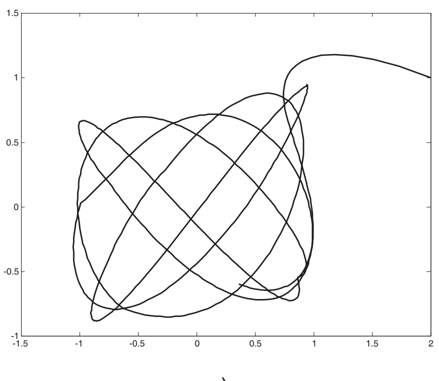
a) b)

Fig. 3: State x_1 a) without the presence of the noise, no barrier function, b) with noise added, no barrier function

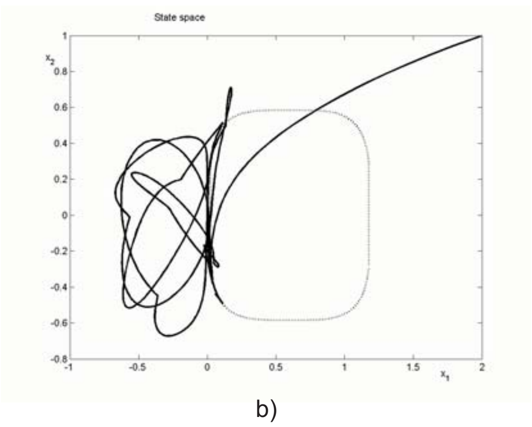


a) b)

Fig. 4: State x_1 a) without presence of the noise, with barrier function, b) with noise and barrier function added



a)



b)

Fig. 5: State space a) no barrier active, b) with the barrier active

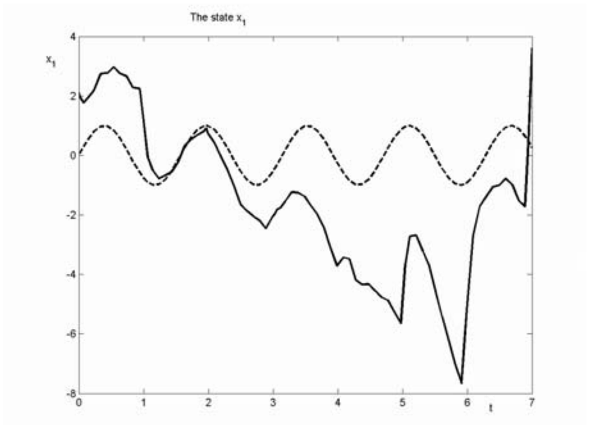


Fig. 6: State x_1, directly controlled by the upper level

7 Remarks

If the penalty on the tracking error is too great ($q_i = 1000$), certain problems with satisfying the boundary conditions occur – these conditions are not satisfied even after a great number of iterations. Using such an inaccurate approximation of the optimal trajectory significantly decreases the quality of the tracking.

Other problems occur if the barrier function is too steep – the computations often end with an error message.

It is necessary to consider the huge time consumption in the upper level. This is due to the feedforward computation. At this stage a differential equation is solved. The optimal trajectory x is on the right-hand side of this equation. This results in the need to call the Femlab function **postinterp** always when the right-hand side is evaluated. The postprocessing seems to be rather time-consuming, and in this case its time consumption of the time exceeds considerably exceeds the time necessary for solving the boundary-value problem. This difference is just strengthened by the fact that the optimal trajectory need not be evaluated with high precision.

8 Conclusion

We have designed and simulated a hierarchical controller where the optimization is based on the solution of a boundary control problem. This problem is solved in Femlab 2.1. The solution of this problem represents the reference trajectory for the lower level of the hierarchical controller.

Acknowledgment

This work was partially supported by research program No J04/98:212300013 "Decision Making and Control for Manufacturing" of the Czech Technical University in Prague (sponsored by the Ministry of Education, Youth and Sports of the Czech Republic), and by grant No 102/01/1347 of the Grant Agency of the Czech Republic.

References

[1] Bertsekas, D. P.: *Dynamic Programming and Optimal Control*. Belmont: Athena Scientific, 1995.

[2] Bryson, A. E., Ho, Y.: *Applied Optimal Control*. New York: J. Wiley, 1975.

[3] *Femlab Reference Manual v. 2.0*, Stockholm: COMSOL AB, 2000

[4] Jamshidi, M.: *Large Scale Systems: Modeling, Control and Fuzzy Logic*. New Jersey: Prentice Hall, 1997.

[5] Maciejowski, J. M.: *Predictive Control with Constraints*. Harlow: Prentice Hall, 2002.

[6] Rehák, B.: "Design of a Hierarchical Controller Using Femlab." In: Proceedings of the 22nd IASTED Conference on Modelling, Identification and Control (Editor: M. H. Hamza), Anaheim, ACTA Press, p. 543–547.

[7] Singh, M. G.: *Decentralised Control*. Amsterdam: North Holland, 1981.

[8] Trave, L., Titli, A., Tarras, A.: *Large Scale Systems: Decentralization, Structure Constraints and Fixed Modes*. Berlin: Springer, 1989.

[9] Vincent, T. L., Grantham, J.: *Nonlinear and Optimal Control Systems*. New York: J. Wiley, 1997.

Ing. Mgr. Branislav Rehák
phone: +420 2 2435 7336
fax: +420 2 2491 864
e-mail: rehakb@control.felk.cvut.cz

Department of Control Engineering

Czech Technical University in Prague
Faculty of Electrical Engineering
Technická 2
166 27 Prague 6, Czech Republic