Nonlinear Model Predictive Control of a Robot Manipulator

POIGNET Ph.*, GAUTIER M.**

*Laboratoire de Vision et Robotique de Bourges, U.P.R.E.S. E.A. 2078
I.U.T. de Bourges, 63 Ave de Lattre de Tassigny
18 020 Bourges
**Institut de Recherche en Cybernétique de Nantes (IRCyN),U.M.R. C.N.R.S. 6597
1 rue de la Noë, BP 92101
44321 Nantes Cedex 03, FRANCE

Corresponding author : Ph. Poignet, Phone : (33) 2 48 23 80 50, Fax : (33) 2 48 23 80 51
e-mail : philippe.poignet@bourges.univ-orleans.fr

Abstract : In this paper, an efficient approach for nonlinear model predictive control is proposed. Basically, the model is first linearized by feedback [8], secondly a model predictive control scheme [9], implemented with an optimized dynamic model and running within small sampling period, is exhibited. Major simulation results performed using numerical values of an industrial SCARA type robot prove the effectiveness of the proposed approach. The nonlinear model-based predictive control and the commonly used computed torque control are compared. The tracking performances and the robustness with respect to external disturbances or model / robot mismatch, are enlightened.

1 Introduction

Many strategies have been used in the last few years for robot control [1]. But in the past decade model predictive control (MPC) has become a good control strategy for a large number of process [2]. Several works [3], [4], [5] have shown that model based predictive control could be of a great interest for handling constrained processes by optimizing, over some manipulated inputs, forecasts of process behavior. It provides good performances in term of rapidity, disturbances and errors cancellations [6]. However, a significant number of industrial applications can be found, for instance, in chemicals industries or food processing that is processes with relatively slow dynamics. But very few results concern the computation of the model predictive control with nonlinear and high dynamic process such as robot links where the model is highly nonlinear and the digital control has to be performed within only few milliseconds. In [7], Richalet and al. have implemented their predictive controller on a KUKA robot but they only use an approximated linear model of the robot.

Therefore, the contribution of this work is to propose an efficient approach for nonlinear model predictive control of a *m*-link industrial robot manipulator. Basically, the procedure will consist in two steps i) the model is first linearized by feedback, since it can be proved that the state equation of a *m*-link robot is input-state linearizable [8] ii) secondly the model predictive control scheme developed by Richalet [3], [7], [9] and fully adapted for the multivariable constrained case is computed on the robot dynamic model.

The paper is organized as follows : section 2 recalls the basis of the nonlinear dynamic model of an *n*-link industrial robot, section 3 details the model predictive functional control, section 4 introduces the compared control scheme that is the nonlinear model-based predictive control and the commonly used computed torque control (both of them implemented with an optimized dynamic model) and finally section 5 exhibits major simulation results running within small sampling period and focus on a 2-axis SCARA type robot. The simulations will be performed using numerical values of an industrial SCARA type robot in the presence of actuator saturations. Finally the tracking performances and the robustness with respect to external disturbances or model / robot mismatch, will be enlightened.

2 Dynamic robot modeling

2.1 Inverse dynamic model

The inverse dynamic model of a rigid robot composed of *n* moving links calculates the motor torque vector Γ (the control input) as a function of the generalized coordinates q, $\dot{q}$, $\ddot{q}$ (the state vector of the joint variable and its derivative). It can be obtained from the Lagrangian equation as recalled here [1]:

$$\Gamma = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} + \Gamma_f \qquad (1)$$

where :

q, $\dot{q}$ are the n-dimensional vectors of generalized joint positions and velocities,

L is the Lagrangian of the system, equal to $E(q,\dot{q}) - U(q)$,

E and U are the kinetic and potential energy of the system with :

$$E = \frac{1}{2}\dot{q}^T M(q)\,\dot{q} \qquad (2)$$

0-7803-5976-3/00/$10.00 ©2000 IEEE.

$M(q)$ is the (n * n) positive definite robot inertia matrix.

Γ_f is the friction torque which is usually modeled at non zero velocity as following :

$$\Gamma_{f_j} = F_{s_j} \mathrm{Sign}(\dot{q}_j) + F_{v_j} \dot{q}_j \quad (3)$$

$\dot{q}_j$ is the joint j velocity,

Sign(x) denotes the sign function,

F_{v_j}, F_{s_j} are the viscous and Coulomb friction coefficients of joint j.

Eq.1 can be written as the classical inverse dynamic model which explicitly depends on the joint acceleration :

$$\Gamma = M(q)\,\ddot{q} + H(q, \dot{q}) \quad (4)$$

with :

$$H(q, \dot{q}) = \dot{M}(q)\,\dot{q} - \frac{\partial E}{\partial q} + \frac{\partial U}{\partial q} + \Gamma_f \quad (5)$$

$\ddot{q}$ is the n-dimensional vector of joint accelerations.

$H(q, \dot{q})$ is the n-dimensional vector of centrifugal, Coriolis, gravitational and friction torques.

2.2 Direct dynamic model

The direct dynamic model, which is used for simulation, calculates the joints accelerations as a function of $q, \dot{q}$ and the input torque or forces Γ_j of the motors applied to the joints and the external forces applied on the robot links :

$$\ddot{q} = f(q, \dot{q}, \Gamma) \quad (6)$$

To compute the direct dynamic model, two classes of methods are generally used :

i) At first calculate the inverse dynamic model Eq. (4) using a recursive Newton-Euler algorithm and numerically solve the implicit non linear differential equation :

$$M(q)\,\ddot{q} = \Gamma - H(q, \dot{q}) \quad (7)$$

Because of the definite positiveness of $M(q)$, the Cholesky factorization is very suitable and numerically robust for this purpose.

ii) Use the Featherstone algorithm to calculate a symbolic solution of Eq.(7) [10].

These nonlinear model depend on the dynamic parameters composed of the inertial parameters (masses, first moments and inertia elements) and the friction parameters. Given the robot geometric parameters and its dynamic parameters, the geometric, kinematic and dynamic models are calculated automatically by the SYMORO (SYmbolic MOdelling of RObots) software package [11]. This software provides the different models in an optimized form under C or Matlab code useable directly in the MATLAB / SIMULINK environment.

3 Predictive Functional Control

This section is dedicated to briefly recall the main steps of the model predictive functional control scheme used hereafter. This predictive technique has been developed by Richalet and complete details of the computation may be found in [3], [7], [9].

3.1 Internal Modeling

The model used is a linear one given by :

$$\begin{aligned} x_M(n) &= F_M x_M(n-1) + G_M u(n-1) \\ y_M(n) &= C_M^T x_M(n) \end{aligned} \quad (8)$$

where :

x_M is the state,

u is the input,

y_M is the measured model output,

F_M, G_M and C_M are respectively matrices or vectors (or matrices) of the right dimension.

This model is called an independent model because it is only fed with the measured process inputs (Figure 1).

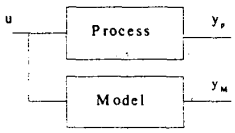

Figure 1 : *Independent Model*

Remark : the problem of robustness because of the poles cancellation by the controller in case of an independent model and if the system is unstable is usually solved by a model decomposition [9].

3.2 Reference trajectory

The predictive control strategy of the MPC is summarized on Figure 2. Given the set point trajectory on a receding horizon $[0, h]$, the predicted process output $\hat{y}_p$ will reach the future set point following a reference trajectory y_R.

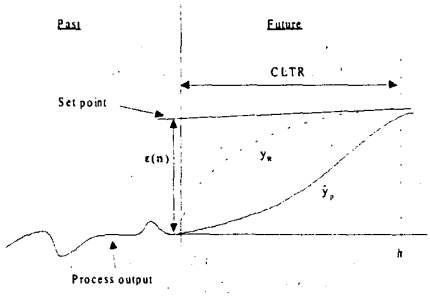

Figure 2 : *Reference Trajectory and predictive control strategy*

where :

$\varepsilon(n) = c(n) - y_p(n)$ is the position tracking error at time n ,

c is the set point trajectory,

y_p is the process output,

CLTR is the closed loop time response.

On the prediction horizon, the reference trajectory y_R, which is the path towards the future set point, is given by :

$$c(n+i) - y_R(n+i) = \alpha^i (c(n) - y_p(n)) \text{ for } 0 \le i \le h \quad (9)$$

where α $(0 \prec \alpha \prec 1)$ is a scalar which has to be chosen in function of the desired closed loop response time.

The predictive essence of the control strategy is completed included in Eq. 9. Indeed, the objective is to track the set point trajectory following the reference trajectory. The reference trajectory may be considered as the desired closed loop behavior.

3.3 Performance index

The performance index may be a quadratic sum of the errors between the predicted process output $\hat{y}_p$ and the reference trajectory y_R. It is defined as follows :

$$D(n) = \sum_{j=1}^{n_h} \left\{ \hat{y}_p(n+h_j) - y_R(n+h_j) \right\}^2 \quad (10)$$

where :
n_h is the number of coincident time point,
h_j are the coincidence time point on the prediction horizon.

The predicted output $\hat{y}_p$ is usually defined as :

$$\hat{y}_p(n+i) = y_M(n+i) + \hat{e}(n+i) \quad 1 \le i \le h \quad (11)$$

where :
y_M is the model output,
$\hat{e}$ is the predicted future output error.

It may be convenient to add a smoothing control term in the performance index. The index becomes :

$$D(n) = \sum_{j=1}^{n_h} \left\{ \hat{y}_p(n+h_j) - y_R(n+h_j) \right\}^2 + \lambda \left\{ u(n) - u(n-1) \right\}^2 \quad (12)$$

where u is the control variable.

3.4 Control variable

The future control variable is assumed to be composed of a priori known functions :

$$u(n+i) = \sum_{k=1}^{n_B} \mu_k(n) u_{Bk}(i) \quad 0 \le i \le h \quad (13)$$

where :
μ_k are the coefficients to be computed during the optimization of the performance index,
u_{Bk} are the base functions of the control sequence,
n_B is the number of base functions.
The choice of the base functions depends on the nature of the set point and the process. Hereafter we will use :

$$u_{Bk}(i) = i^{k-1} \quad \forall k \quad (14)$$

In fact, the only first term is effectively applied for the control, that is :

$$u(n) = \sum_{k=1}^{n_u} \mu_k(n) u_{Bk}(0) \quad (15)$$

The model output is composed two parts :

$$y_M(n+i) = y_{UF}(n+i) + y_F(n+i) \quad 1 \le i \le h \quad (16)$$

where :
y_{UF} is the free output response $(u = 0)$,
y_F is the forced output response to the control variable given by Eq. 13.

Given Eq. 8 and Eq. 13 , it follows :

$$\begin{aligned} y_{UF}(n+i) &= C_M^T F_M^i x_M(n) \quad 1 \le i \le h \\ y_F(n+i) &= \sum_{k=1}^{n_u} \mu_k(n) y_{Bk}(i) \quad 0 \le i \le h \end{aligned} \quad (17)$$

where y_{B_k} is the model output response to u_{B_k}.
Assuming that the predicted future output error is approximated by a polynomial, it follows :

$$\hat{e}(n+i) = e(n) + \sum_{m=1}^{de} e_m(n) i^m \quad \text{for} \quad 1 \le i \le h \quad (18)$$

where :
de is the degree of the polynomial approximation,
e_m are some coefficients calculated on line knowing the past and present output error.

The minimization of the performance index without smoothing control term, in the case of the polynomial base functions, leads to the applied control variable :

$$u(n) = k_0 \left\{ c(n) - y_p(n) \right\} - \sum_{m=1}^{de} k_m e_m(n) + v_x^T x_M(n) \quad (19)$$

where the gains k_0, k_m, v_x^T are calculated off-line (see Appendix).

Therefore the control variable is composed of three terms : the first one is due to the tracking position error, the second one is placed especially for disturbance rejection and the last one corresponds to a model compensation.

4 Compared control strategies

4.1 Feedback linearization

Before applying control strategies, it is basically interesting to convert the nonlinear dynamic model of robot into a linear one. Let's consider the nonlinear dynamic equations for an *m*-link robot expressed as follows (see section 2) :

$$\Gamma = M(q)\ddot{q} + H(q,\dot{q}) \quad (20)$$

It is well known that the rigid *m*-link robot equations may be linearized and decoupled by nonlinear feedback [8]. In fact, given the state space vector and the selected output :

$x_1 = q,\ x_2 = \dot{q}, x=\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ and $y = x_1$,

Eq. 6 can be written as follows :

$$\dot{x}=Ax+B\beta^{-1}(x)[\Gamma-\alpha(x)] \tag{21}$$

where :

$$A=\begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix},\ B=\begin{bmatrix} 0 \\ I \end{bmatrix},\ \beta(x) = M(x_1) \tag{22}$$

and

$$\alpha(x) = H(x_1, x_2) \tag{23}$$

Considering a nonlinear feedback given by :

$$\Gamma=\alpha(x)+\beta(x)v \tag{24}$$

The transfer between v and y is equivalent to :

$$\ddot{y}=v \tag{25}$$

This is known as the feedback linearized system (Figure 3). It corresponds to the familiar inverse dynamics control scheme which transforms Eq. 6 into a double set of integrator equations.

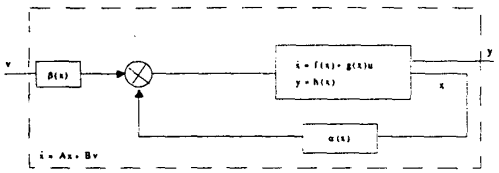

Figure 3 : *Feedback linearized system*

Classical linear system control techniques can now be used to design a tracking controller. In the next sections, we will then compare the efficiency of the model predictive control described in section 3 and a classical computed torque control.

4.2 Computed torque control

Assuming that the motion is completely specified with the desired position (q^d), velocity ($\dot{q}^d$) and acceleration ($\ddot{q}^d$), the classical computed torque control [1] computes the required arm torque in the following manner :

$$v=K_p(q^d-q)+K_v(\dot{q}^d-\dot{q})+\ddot{q}^d \tag{26}$$

where K_p, K_v are the controller gains.

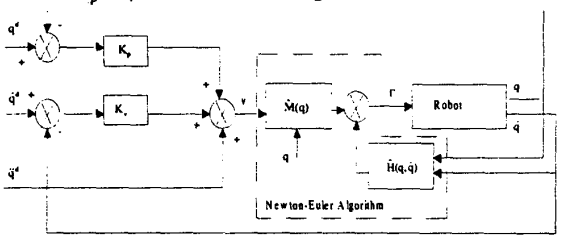

Figure 4 : *Computed Torque Control*

Figure 4 illustrates its implementation for simulation. The gains tuning leads to $K_{P_1}=25$, $K_{V_1}=10$, $K_{P_2}=900$ and $K_{V_2}=60$ in order to guarantee the torque and dynamic actuator constraints in tracking situation as well as in disturbance rejection.

4.3 Predictive functional control

The MPC algorithm is implemented with an internal model composed of a double set of integrator. Three different base functions are used : step, ramp and parabola. The close loop response time is fixed to $CLTR = 9 * T_{sampling}$ in order to ensure the compromise between the tracking performances and the disturbance rejection. Three coincidence time points on the prediction horizon are defined :
$h_1 = 1/3CLTR/T_{sampling}$, $h_2 = 2/3CLTR/T_{sampling}$, $h_3 = CLTR/T_{sampling}$.
The predicted future output error is approximated by a polynomial of degree 2.

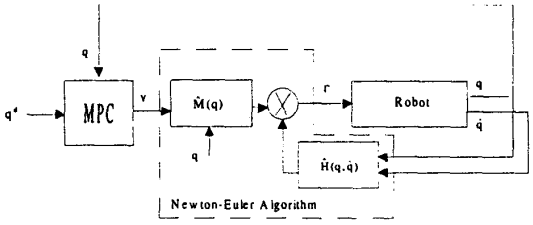

Figure 5 : *Model Predictive Control*

5 Simulation results

5.1 Description of the SCARA robot

The comparison is carried out on a 2 joints planar direct drive prototype robot (Figure 1), without gravity effect. The robot is direct driven by 2 DC permanent magnet motors supplied by PWM choppers. The description of the geometry of the robot uses the modified Denavit and Hartenberg notation [12,13,14].

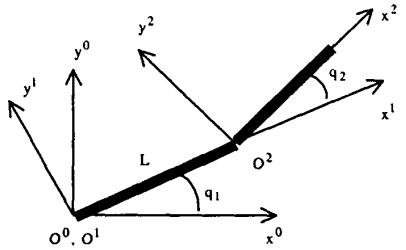

Figure 6 : *SCARA robot, frames and joint variables.*

The choice of modified Denavit and Hartenberg frames attached to each body allows to obtain a dynamic model linear in relation to a set of standard dynamic parameters Θ_S :

$$\Gamma=D_S(q, \dot{q}, \ddot{q})\,\Theta_S \tag{27}$$

The (13nx1) vector Θ_s is composed for each link of the 6 components of the inertia tensor, the 3 components of the first moment and the mass, a total inertia moment for rotor actuator and gears, Coulomb and viscous friction parameters. It has been shown that the set of standard dynamic parameters can be simplified to obtain the base inertial parameters [13]. In the case considered here, the dynamic model depends on 8 minimal dynamic parameters, considering 4 friction parameters :

$$\Theta=[ZZR_1 \quad Fv_1 \quad Fs_1 \quad ZZ_2 \quad MX_2 \quad MY_2 \quad Fv_2 \quad Fs_2]^T \quad (28)$$

where :

$ZZR_1 = ZZ_1 + M_2L^2$,

L is the length of the first link,

M_2 is the mass of the link 2,

ZZ_1 and ZZ_2 are the drive side moment of inertia of links 1 and 2 respectively,

MX_2 and MY_2 are the first moments of link 2.

5.2 Inverse dynamic model

The inverse dynamic model is given by :

$$\begin{bmatrix}\Gamma_1\\ \Gamma_2\end{bmatrix} = \begin{bmatrix} \ddot{q}_1 & 0 \\ \dot{q}_1 & 0 \\ sign(\dot{q}_1) & 0 \\ \ddot{q}_1+\ddot{q}_2 & \ddot{q}_1+\ddot{q}_2 \\ (2\ddot{q}_1+\ddot{q}_2)C2-\dot{q}_2(2\dot{q}_1+\dot{q}_2)S2 & \ddot{q}_1C2+\dot{q}_1^2S2 \\ -(2\ddot{q}_1+\ddot{q}_2)S2-\dot{q}_2(2\dot{q}_1+\dot{q}_2)C2 & \dot{q}_1^2C2-\ddot{q}_1S2 \\ 0 & \dot{q}_2 \\ 0 & sign(\dot{q}_2) \end{bmatrix}^T \Theta \quad (29)$$

where : $C2 = \cos(q_2)$ and $S2 = \sin(q_2)$.

The inverse and direct dynamic models computed by SYMORO+ [8] are implemented using optimized algorithm.

5.3 Identification

The parameters are estimated using weighted least square techniques. More details may be found in [12,13,14,15]. The estimated values, given in Table 1, will be considered as the nominal values during the simulations.

Physical Parameters	Estimated values
ZZR1	3.38
Fv1	0.1
Fs1	0.57
ZZ2	0.063
MX2	0.242
MY2	0.001
Fv2	0.012
Fs2	0.125

Table 1 : *Parameters of the simulated robot manipulator*

5.4 Tracking performances

These tests are running within 6ms sampling period. Hereafter CTC stands for Computed torque control and PFC for Predictive Functional Control.

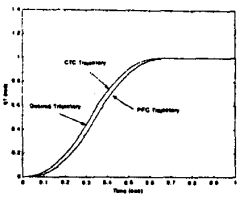

Figure 7 : *Measured and desired trajectories (q_1)*

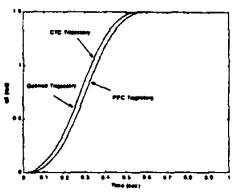

Figure 8 : *Measured and desired trajectories (q_2)*

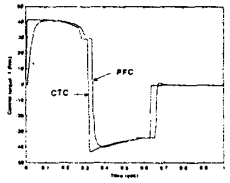

Figure 9 : *Control torque 1*

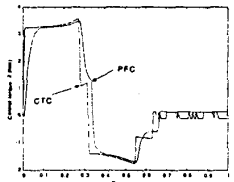

Figure 10 : *Control torque 2*

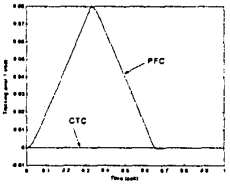

Figure 11 : *Tracking error 1*

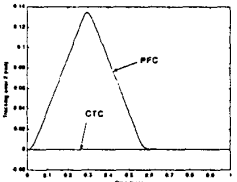

Figure 12 : *Tracking error 2*

The tracking performances are not very advantageous in this situation with respect to the CTC control, which can also be considered in this manner as a predictive tracking control. But the interest of the PFC control will appear in the next subsections concerning the disturbance rejection and the model mismatch.

5.5 Disturbance rejection

The applied unknown output disturbance is constant and equal to 0.05rad for q_1 and 0.1rad for q_2.

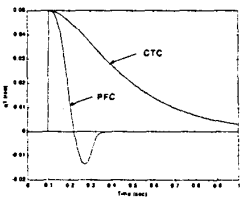

Figure 13 : *Disturbance rejection on q1*

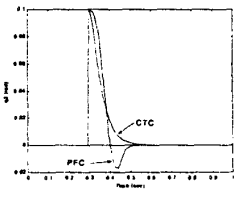

Figure 14 : *Disturbance rejection on q2*

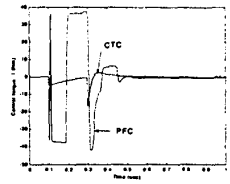

Figure 15 : *Control Torque 1*

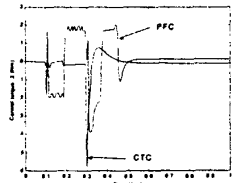

Figure 16 : *Control Torque 2*

With the same controller tuning, the PFC control is very robust with respect to external disturbances. The rejection time is much smaller for the first axis. In this case, adding a

constraint on the PFC control variable magnitude leads to efficient results.

5.6 Model mismatch robustness

The model mismatch (Table 2) is obtained considering up to 50% errors from the nominal values given in Table 1. The desired trajectories are those of section 5.4.

ZZR1	5.07	FV1	0.15
ZZ2	0.0945	FV2	0.018
MX2	0.363	FS1	0.855
MY2	0.015	FS2	0.1875

Table 2 : *Mismatched parameters*

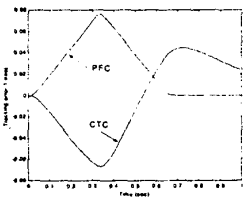

Figure 17 : *Tracking error 1*

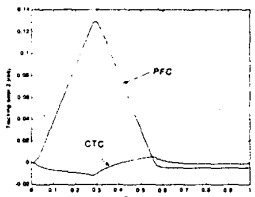

Figure 18 : *Tracking error 2*

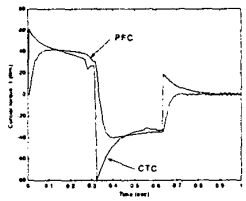

Figure 19 : *Control torque 1*

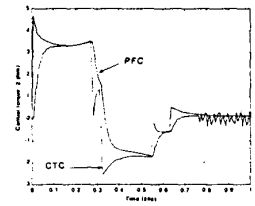

Figure 20 : *Control torque 2*

The PFC approach is robust to model mismatch. Up to 50% of modeling error, the behavior is very satisfactory.

6 Conclusion

This paper exhibits fully the tracking performances of the PFC and its robustness especially in the case of disturbance rejection and model mismatch, with a control scheme using the inverse dynamic model. Further works will concern the implementation of this predictive control scheme computed with the inverse dynamic model on a SCARA type robot.

7 References

[1] CANUDAS de WIT C., SICILANO B., BASTIN G., *Theory of Robot Control,* Springer Verlag, 1996.

[2] ALLGÖWER F., BADGWELL T.A., QIN J.S., RAWLINGS J.B., WRIGHT S.J., *Nonlinear Predictive Control and Moving Horizon Estimation - An Introductory Overview,*ECC'99, pp. 391-449.

[3] RICHALET J., RAULT A., TESTUD J. & PAPON J., *Model Predictive Heuristic Control : Applications to Industrial Process,* Automatica, Vol. 14, 1978.

[4] CLARKE D.W., MOHTADI C. & TUFFS P.S., 1987, *Generalised Predictive Control. Part. 1 : The Basic Algorithm. Part. 2 : Extensions and Interpretations,* Automatica, Vol. 23, n°2, pp. 137-160.

[5] BOUCHER P., DÜMUR D. & RAHMANI FAISSAL K., *Generalised Predictive Cascade Control (GPCC),*Proceeding of ECC'91, pp. 734-739, Grenoble, July 1991.

[6] BOUCHER P., DÜMUR D. & RAHMANI FAISSAL K., *Predictive Cascade Control of Machine Tools Motor Drives,* Proceeding of EPE'91, Florence, Vol. 2, pp. 120-125, Sept. 1991.

[7] RICHALET J., ABU El Ata-Doss S., ARBER C., KUNTZE H.B., JACUBASCH A., SCHILL W., *Predictive Functional Control. Application to Fast and Accurate Robots,* 10th IFAC World Congress, Munich, 1997.

[8] KHALIL H.K., *Non Linear Systems,* Prentice Hall, 2nd Edition. 1996.

[9] RICHALET J., *Pratique de la Commande Prédictive,* Hermès, 1993.

[10] FEATHERSTONE R., *The calculation of robot dynamics using articulated-body inertias,* The Int. J. of Robotics Research, Vol. 2(3), 1983, pp. 87-101.

[11] KHALIL W., *SYMORO+ : A system for generating the symbolic models of robots,* SYROCO, Capri, Sept. 1994.

[12] GAUTIER M., W. KHALIL, *On the identification of the inertial parameters of robots,* 27th CDC, 1988, pp. 2264-2269.

[13] GAUTIER M., KHALIL W., *Direct calculation of minimum inertial parameters of serial robots,* IEEE Trans. on Robotics and Automation, Vol. 6, N° 3, 1990, pp. 368-373.

[14] GAUTIER M., *Dynamic identification of robots with power model,* Int. Conf. On Robotics and Automation, Albuquerque, April 1997.

[15] POIGNET Ph., GAUTIER M., *Comparison of weighted least squares and extended kalman filtering methods for dynamic identification of robots,* Int. Conf. On Robotics and Automation, San Francisco, April 2000, to be appeared.

8 Appendix

$$k_0 = v^T \begin{pmatrix} 1-\alpha^{h_1} \\ 1-\alpha^{h_2} \\ \vdots \\ 1-\alpha^{h_{n_h}} \end{pmatrix},\quad k_m = v^T \begin{pmatrix} h_1^m \\ h_2^m \\ \vdots \\ h_{n_h}^m \end{pmatrix},\quad v_x = -\begin{bmatrix} C_M^T(F_M^{h_1} - I) \\ C_M^T(F_M^{h_2} - I) \\ \vdots \\ C_M^T(F_M^{h_{n_h}} - I) \end{bmatrix}^T v \qquad (30)$$

$$v = R^T u_B(0) \qquad (31)$$

where :

$$R = \left\{ \sum_{j=1}^{n_h} y_B(h_j) y_B(h_j)^T \right\}^{-1} \left[y_B(h_1) \quad y_B(h_2) \quad \cdots \quad y_B(h_{n_h}) \right] \qquad (32)$$

$$y_B = \left(y_{B_1} \quad y_{B_2} \quad \cdots \quad y_{B_{n_B}} \right)^T \qquad (33)$$

$$u_B = \left(u_{B_1} \quad u_{B_2} \quad \cdots \quad u_{B_{n_B}} \right)^T \qquad (34)$$