PREDICTIVE COMPUTED-TORQUE CONTROL OF A PUMA 560 MANIPULATOR ROBOT

Victor M. Becerra, Steven Cook and Jiamei Deng

University of Reading, Department of Cybernetics, Reading RG6 6AY, UK

Abstract: This paper describes the integration of constrained predictive control and computed-torque control, and its application on a six degree-of-freedom PUMA 560 manipulator arm. The real-time implementation was based on SIMULINK, with the predictive controller and the computed-torque control law implemented in the C programming language. The constrained predictive controller solved a quadratic programming problem at every sampling interval, which was as short as 10 ms, using a prediction horizon of 150 steps and an 18th order state space model.

Keywords: Predictive control, robotic manipulators, computed-torque control.

1. INTRODUCTION

This paper describes the integration of constrained predictive control and computed-torque control, and its application in real time to control a six degree-of-freedom PUMA 560 manipulator arm. Until recent years the application of constrained predictive control to a manipulator with six degrees of freedom, although theoretically possible, has not been practical as computer processors have not been fast enough to solve online the associated constrained optimisation problem within the short sampling periods that are required by the application. Predictive control has been proposed on a simulated two-link PUMA 560 arm as described in (Torres *et al.*, 2001), who used local linearisation to define the internal model. In the work by (Bemporad *et al.*, 1997), a predictive path generator was designed to deal with various constraints, and experiments on a PUMA 560 manipulator were made using three links of arm.

The essence of predictive control is to optimise, over the manipulable inputs, forecasts of process behaviour (Maciejowski, 2002). One of the main benefits of predictive control is that constraints on the inputs and outputs of the system can be explicitly considered in the control problem formulation and its solution.

The success of linear predictive control has inspired researchers to look into the possibility of extending it for non-linear control applications. Since manipulator robots exhibit strong nonlinearities, their performance can be significantly improved by using nonlinear control strategies. Computed-torque control (Lewis *et al.*, 2004) is a technique that uses a nonlinear dynamic model of the system to remove the nonlinearities of the manipulator, facilitating external control with fixed gains. Poignet *et al.* (2000) describe a combined predictive functional control / computed-torque control scheme, with simulated tests on a two degree-of-freedom SCARA robot.

The PUMA 560, shown in Figure 1, is a six-degree of freedom robotic manipulator that uses six dc servomotors for joint control. Joint positions are measured using encoders and potentiometers. Three large motors provide control of

the waist, shoulder, and elbow, while three smaller motors position the orientation of the wrist. The PUMA 560 has a large reach, and can achieve impressive acceleration. Originally designed for assembly and manipulation tasks, the PUMA arm is now widely adopted by academic institutions for research purposes.

Fig. 1. PUMA robot manipulator arm

2. DESCRIPTION OF THE CONTROL SCHEME

2.1 Computed torque control (CTC)

The reference torque for each joint of the arm is calculated using computed-torque control (Lewis *et al.*, 2004). This technique is used to remove the nonlinearities of the PUMA by employing feedback linearisation. The arm dynamics are given by:

$$M(q)\ddot{q} + N(q,\dot{q}) + \tau_d = \tau \tag{1}$$

where $q(t) \in \Re^6$ is a vector of joint variables, $\tau(t) \in \Re^6$ the control torque, $\tau_d(t) \in \Re^6$ is a disturbance, $M(q)$ is the inertia matrix, $N(q,\dot{q})$ represents nonlinear terms including coriolis/centripetal effects, friction, and gravity.

Suppose that a reference trajectory $q_d(t)$ has been chosen for the arm motion. The tracking error is defined as:

$$e(t) = q_d(t) - q(t) \tag{2}$$

If the tracking error is differentiated twice, then

$$\ddot{e} = \ddot{q}_d + M^{-1}(N + \tau_d - \tau). \tag{3}$$

Define the feedback linearising input function as

$$u = \ddot{q}_d + M^{-1}(N - \tau), \tag{4}$$

and the disturbance function as

$$w = M^{-1}\tau_d. \tag{5}$$

Then the tracking error dynamics can be expressed as follows

$$\frac{d}{dt}\begin{bmatrix} e \\ \dot{e} \end{bmatrix} = \begin{bmatrix} 0 & I_{6\times 6} \\ 0 & 0 \end{bmatrix}\begin{bmatrix} e \\ \dot{e} \end{bmatrix} + \begin{bmatrix} 0 \\ I_{6\times 6} \end{bmatrix} u + \begin{bmatrix} 0 \\ I_{6\times 6} \end{bmatrix} w. \tag{6}$$

Notice that, as a result of using the feedback linearising transformation (4), the tracking error dynamics are given by a linear state equation with constant coefficients. The feedback linearising transformation (4) can be inverted to give

$$\tau = M(q)(\ddot{q}_d - u) + N(q,\dot{q}). \tag{7}$$

This is the computed-torque control law. An outer loop controller is often used. The role of the outer loop controller is to provide the input u. For example, PD and PID and LQR computed-torque controllers have been proposed as outer loop controllers (Lewis *et al.*, 2004). The computed-torque control technique is known to perform well when the robot arm parameters are known fairly accurately. Fortunately, the dynamics of the PUMA 560 manipulator are well known and reported. The inverse dynamics and Denavit-Hartenberg arm parameters employed in this work are based on those reported in (Lewis *et al.*, 2004) and (Corke and Armstrong-Helouvry, 1994).

2.2 The predictive control problem

In this work, a constrained predictive controller is employed as an outer loop controller to generate the input u, and the robot dynamics are feedback linearised by means of the computed-torque control law (7). The predictive control formulation below assumes a linear, discrete-time, state-space model of the plant:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= C_y x(k) \\ z(k) &= C_z x(k), \end{aligned} \tag{8}$$

where $x(k)$ is the state vector at time k, $u(k)$ is the vector of inputs, $y(k)$ is the vector of measured outputs, and $z(k)$ is the vector of outputs which are to be controlled to satisfy some constraints, or to particular set-points, or both. In this work, a Kalman filter was used that can be described as follows:

$$\begin{aligned} \hat{x}(k+1|k) &= A\hat{x}(k|k-1) + Bu(k) + L\hat{e}(k|k) \\ \hat{y}(k|k-1) &= C_y\hat{x}(k|k-1) \\ \hat{z}(k|k-1) &= C_z\hat{x}(k|k-1), \end{aligned} \tag{9}$$

where $\hat{x}(k+1|k)$ is the estimate of the state at future time $k+1$ based on the information available at time k, $\hat{y}(k|k-1)$ is the estimate of the plant output at time k based on information at time $k-1$, L is the Kalman filter gain matrix and $\hat{e}(k|k)$ is the estimated error: $\hat{e}(k|k) = y(k) - \hat{y}(k|k-1)$.

The formulation given in this paper is inspired by the constrained predictive control algorithm presented in (Maciejowski, 2002). The cost function V minimised by the predictive controller penalises deviations of the predicted controlled

outputs $\hat{z}(k+i|k)$ from a reference trajectory $r(k+i|k)$ and also it penalises changes in the future manipulated inputs $\Delta\hat{u}(k+i|k)$. Define the cost function as follows

$$V(k) = \sum_{i=1}^{p} ||\hat{z}(k+i|k) - r(k+i|k)||^2_{Q(i)} + \sum_{i=0}^{m-1} ||\Delta\hat{u}(k+i|k)||^2_{R(i)}, \quad (10)$$

where the prediction and control horizons are p and m, respectively, $Q(i)$ and $R(i)$ are output and input weight matrices, respectively. The cost function can also be written as follows:

$$V(k) = ||Z(k) - T(k)||^2_Q + ||\Delta U(k)||^2_R, \quad (11)$$

where

$$Z(k) = \begin{bmatrix} \hat{z}(k+1|k) \\ \vdots \\ \hat{z}(k+p|k) \end{bmatrix}$$
$$T(k) = \begin{bmatrix} \hat{r}(k+1|k) \\ \vdots \\ \hat{r}(k+p|k) \end{bmatrix} \quad (12)$$
$$\Delta U(k) = \begin{bmatrix} \Delta\hat{u}(k|k) \\ \vdots \\ \Delta\hat{u}(k+m-1|k) \end{bmatrix},$$

$$R = \begin{bmatrix} R(0) & 0 & \dots & 0 \\ 0 & R(1) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & R(m-1) \end{bmatrix}$$
$$Q = \begin{bmatrix} Q(1) & 0 & \dots & 0 \\ 0 & Q(2) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & Q(p) \end{bmatrix}. \quad (13)$$

Note that

$$Z(k) = \psi x(k) + \Upsilon u(k-1) + \Theta\Delta U(k), \quad (14)$$

where

$$\psi = \begin{bmatrix} C_zA \\ \vdots \\ C_zA^m \\ C_zA^{m+1} \\ \vdots \\ C_zA^p \end{bmatrix},$$

$$\Upsilon = \begin{bmatrix} C_zB \\ \vdots \\ \sum_{i=0}^{m-1} C_zA^iB \\ \sum_{i=0}^{m} C_zA^iB \\ \vdots \\ \sum_{i=0}^{p-1} C_zA^iB \end{bmatrix},$$

$$\Theta = \begin{bmatrix} C_zB & \dots & 0 \\ AB+B & \dots & 0 \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{m-1} A^iB & \dots & B \\ \sum_{i=0}^{m} A^iB & \dots & AB+B \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{p-1} A^iB & \dots & \sum_{i=0}^{p-m} C_zA^iB \end{bmatrix}.$$

Define

$$\varepsilon(k) = T(k) - \psi x(k) - \Upsilon u(k-1). \quad (15)$$

Now the following can be obtained:

$$\begin{aligned} V(k) &= ||\Theta\Delta U(k) - \varepsilon(k)||^2_Q + ||\Delta U(k)||^2_R \\ &= \varepsilon(k)^TQ\varepsilon(k) - 2\Delta U(k)^T\Theta^TQ\varepsilon(k) \\ &\quad + \Delta U(k)^T[\Theta^TQ\Theta + R]\Delta U(k). \end{aligned} \quad (16)$$

Equation (16) has the form

$$V(k) = \text{const} - \Delta U(k)^T\vartheta + \Delta U(K)^TH\Delta U(k), \quad (17)$$

where

$$\vartheta = 2\Theta^TQ\varepsilon(k), \quad (18)$$

and

$$H = \Theta^TQ\Theta + R. \quad (19)$$

Then, to minimise V as given by equation (10), the following constrained optimization problem can be solved:

$$\min_{\Delta U(k)} \Delta U(k)^TH\Delta U(k) - \vartheta^T\Delta U(k), \quad (20)$$

subject to the inequality constraints:

$$
\begin{aligned}
u(k) &\geq u_{min}(k) \\
u(k) &\leq u_{max}(k) \\
|\Delta u(k)| &\leq \Delta u_{max}(k) \\
z(k) &\geq z_{min}(k) \\
z(k) &\leq z_{max}(k).
\end{aligned} \tag{21}
$$

This optimisation problem can be expressed as follows:

$$
\min_{\theta} \theta^T \phi \theta + \varphi^T \theta, \tag{22}
$$

subject to

$$
\Omega \theta \leq \omega, \tag{23}
$$

where $\theta = \Delta U(k)$, $\phi = H$, $\varphi = -\vartheta$. Expressions for matrix Ω and vector ω are obtained from Equations (21) and (14).

2.3 The state space model employed

The linear tracking dynamics (6) can be discretised using the zero order hold method with sampling interval h to yield, ignoring the unmeasured disturbance input w, the following discrete time model:

$$
\begin{aligned}
x_d(k+1) &= A_d x_d(k) + B_d u(k) \\
y(k) &= C_d x_d(k)
\end{aligned} \tag{24}
$$

where k is an integer time index, $x_d(k) = [e(kh)\, \dot{e}(kh)]^T$ is the discrete time state vector, $y(k) = e(kh)$ is the output vector, and A_d, B_d, C_d, D_d are discrete time model matrices. Define

$$
\begin{aligned}
\Delta u(k) &= \quad u(k) - u(k-1) \\
\Delta x(k) &= \quad x(k) - x(k-1).
\end{aligned} \tag{25}
$$

A convenient augmented model can be obtained as follows:

$$
\begin{aligned}
x(k+1) &= Ax(k) + B\Delta u(k) \\
y(k) &= Cx(k)
\end{aligned} \tag{26}
$$

where

$$
\begin{aligned}
x(k) &= \begin{bmatrix} \Delta x_d(k) \\ y(k) \end{bmatrix} \\
A &= \begin{bmatrix} A_d & 0_{12\times 6} \\ C_d A_d & I_6 \end{bmatrix} \\
B &= \begin{bmatrix} B_d \\ C_d B_d \end{bmatrix} \\
C &= [0_{12\times 12}\, I_6]
\end{aligned} \tag{27}
$$

This augmented 18th order model was used by the real-time predictive controller employed in this work. If the system is at steady state, then $\Delta x(0) = 0$. As $y(0)$ is the initial position of the system, $x(0)$ is known. This augmented model is therefore easily initialized, which is an advantage over the general state space model (24).

3. IMPLEMENTATION AND RESULTS

3.1 Hardware configuration

The original Mark II controller of this PUMA arm has been retrofitted to enable control from a personal computer. Special purpose TRC041 retrofit cards (Mark V Automation Inc., 2000) are installed in the backplane of the Mark II controller. Custom-made cables are used to interface the TRC041 cards and a Q8 data acquisition board (Quanser Consulting Inc., 2003*a*), which is connected to the PCI interface of the personal computer. An Intel Pentium 4 2.4 GHz personal computer running the Windows 2000 operating system is used to control the arm. Servo torques are controlled by the Mark II controller, with reference values sent as analog voltages from the personal computer through the Q8 board. The Q8 board receives encoder and potentiometer signals from the TRC041 and is interfaced with SIMULINK.

3.2 Software configuration

Figure 2 shows a SIMULINK diagram of the real-time control scheme. MATLAB's Real Time Workshop and Wincon (Quanser Consulting Inc., 2003*b*) were used to generate and execute real-time code.

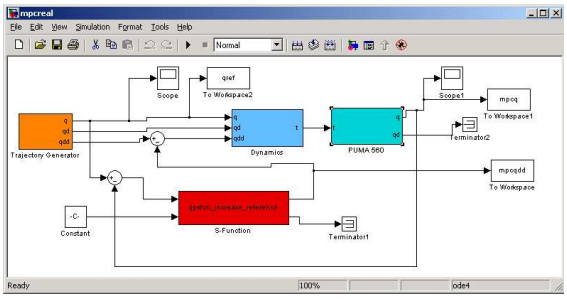

Fig. 2. SIMULINK system implementing the real-time control scheme

The computed torque control law given by Equation (7) was implemented by means of a SIMULINK S-function written in C (The Mathworks, Inc., 2001). The predictive algorithm has been implemented in the C programming language and interfaced to SIMULINK through an S-function. The implemented predictive control algorithm is described in Table 1. The algorithm uses linear state space models of the form given in Equation (26). The algorithm follows the formulation given in section 2.2. A public domain quadratic programming (QP) algorithm by K. Schittkowski was employed (Schittkowski, 1992). The controller tuning parameters can be set through the S-function interface.

Table 1. Predictive control algorithm implemented by S-function

Step	Description
1	read in measured output vector $y(k)$.
2	form matrices ϕ, Ω and vectors φ and ω.
3	solve the quadratic programming problem given by Eqns. (22) and (23).
4	compute the control vector from the resulting increment $u(k) = u(k-1) + \Delta\hat{u}(k\|k)$.
5	store $u(k)$ and increase the sampling index: $k \leftarrow k+1$.
6	write out control signal vector $u(k)$.
7	go to 1 and repeat again.

The software was designed with the assumption that the bounds on variables are constant over their relevant time horizons. Under this assumption, the limits *ulim* and *ylim*, which are manipulated and output variable constraints, respectively, are defined as follows:

$$\begin{aligned} u_{lim} &= [u_{min}^T u_{max}^T]^T \\ y_{lim} &= [z_{min}^T z_{max}^T]^T. \end{aligned} \tag{28}$$

Similarly, it is assumed that the input and output weights are constant along the relevant time horizons, so that $R(1) = R(2) = \cdots R(m) = R$ and $Q(1) = Q(2) = \cdots Q(p) = Q$, with R and Q diagonal matrices. Under this assumption, the weight vectors *uwt* and *ywt*, which are input and output weight vector, respectively, are defined as follows:

$$\begin{aligned} uwt &= [R_{1,1}, \ldots, R_{n_u,n_u}] \\ ywt &= [Q_{1,1}, \ldots, Q_{n_y,n_y}]. \end{aligned} \tag{29}$$

3.3 Constraints

The feedback input to the predictive controller is the tracking error on each joint. The controller output is a correctional value of the angular acceleration which must be subtracted from the required angular acceleration. Constraints can be placed on the input and output of the predictive controller. This is useful since it is, in theory, possible to prevent the manipulator from moving too far away from the reference trajectory. It is also possible to constrain the rate at which the controller can change its input to the system. With the formulation given in this paper, it would be difficult to consider physical constraints such as torque or current, due to the nonlinear connections between the actual torque and servo-motor current, and the predictive controller output.

3.4 Controller tuning

The tuning parameters associated with the predictive controller are the sampling period h, the prediction horizon p, the control horizon m, and the input and output weights *uwt* and *ywt*.

For rapidly changing systems, such as the PUMA arm, a high sample rate is required. A sampling period of about $h = 0.001$ s is usually considered suitable for such a system. It is important to consider the amount of time taken to solve the QP problem, as it must be solved within each sampling period. This load is influenced by factors such as the prediction and control horizon, as well as the size of the model and the speed of the computer. Through experimentation, it was found that the use of a sampling period of $h = 0.01$ s was feasible, as the computer was not fast enough to calculate the solution to the QP with suitable prediction and control horizons within 1 ms.

Initial tuning was carried out based on simultaneous filtered step changes on the references using a simulated manipulator based on the Denavit-Hartenberg model of the PUMA (Lewis *et al.*, 2004), (Corke and Armstrong-Helouvry, 1994). Controller tuning was refined through real-time experiments on the actual manipulator, and the final tuning employed was:

- $p = 150$,
- $m = 6$,
- $uwt = [60, 60, 60, 65, 60, 60]$
- $ywt = [50, 50, 50, 50, 50, 50]$

Figure 3 shows the trajectory of the joint variables q_1 and q_2 of the PUMA under predictive computed-torque control, and the corresponding references, for one of the experiments carried out as part of this work.

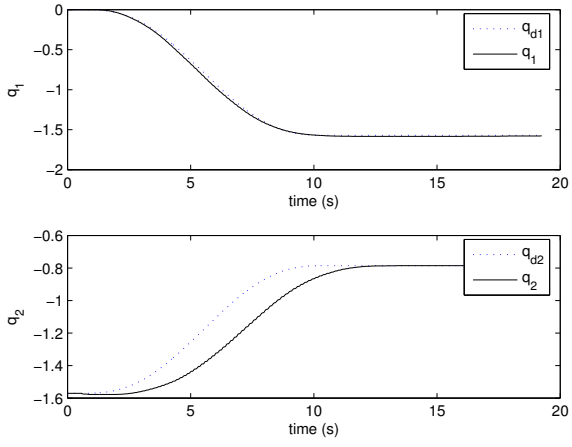

Fig. 3. Trajectory following under predictive computed-torque control.

4. CONCLUSIONS

This paper describes the integration of constrained predictive control and computed-torque

control, and its application in real-time to control a 6-DOF PUMA manipulator arm.

It is felt by the authors that the work reported here is a good start in exploring the benefits of using predictive computed-torque control on the PUMA arm. However, the tuning reported in this paper can be improved. A significant improvement in performance could be achieved with a faster control computer. This is because the sampling period could be made shorter and, conversely, the prediction horizon longer.

Constraints can be placed on the input and output of the predictive controller. This is useful since, in theory, it is possible to prevent the manipulator from moving too far away from the reference trajectory. It is also possible to include information about future reference trajectories in the predictive control computations, such that the controller can optimally anticipate future changes in the reference trajectory. This feature may be useful when the manipulator is required to do repetitive tasks. These capabilities give constrained predictive control a significant advantage over PID computed-torque control which can not optimally handle constraints on input or output variables, or information about future reference trajectories.

5. ACKNOWLEDGEMENTS

This paper summarises the results obtained by Steven Cook in his final year project of the MEng in Cybernetics at the University of Reading. The real-time predictive control algorithm employed in this work was implemented by Jiamei Deng and Victor Becerra with funding from the EPSRC under grant GR/R64193. The computed-torque control algorithm was implemented by Callum Cage as part of his 3rd year project of the MEng in Computer Science and Cybernetics at the University of Reading.

REFERENCES

Bemporad, A., T. Tarn and N. Xi (1997). Predictive path parameterization for constrained robot control. *IEEE Trans. on Control Systems Technology* **7**(6), 648–656.

Corke, P.I. and B. Armstrong-Helouvry (1994). A search for consensus among model parameters reported for the PUMA 560 robot. In: *Proc. IEEE Conf. Robotics and Automation*. pp. 1608–1613.

Lewis, F.L., D.M. Dawson and C.T. Abdallah (2004). *Robot Manipulator Control*. Marcel Dekker.

Maciejowski, J.M. (2002). *Predictive control with constraints*. Prentice Hall.

Mark V Automation Inc. (2000). *TRC004 User's Manual*.

Poignet, P. and M. Gautier (2000). Nonlinear model predictive control of a robot manipulator. In: *Proc. 6th International Workshop on Advanced Motion Control*. Nagoya, Japan. pp. 401 – 406.

Quanser Consulting Inc. (2003*a*). *Q8 Data Acquisition System: WinCon Support and Installation Guide, Version 1.0*.

Quanser Consulting Inc. (2003*b*). *WinCon 4.1 User's Manual, Version 1.2*.

Schittkowski, K. (1992). Quadratic Programming implementation - C version translated from Fortran Version 1.4 (March 1987) - C translation 1992, modified by M.J.D. Powel (University of Cambridge), A.L. Tits, J.L. Zhou and C. Lawrence (University of Maryland). Mathematisches Institut, Universitaet Bayreuth, Germany.

The Mathworks, Inc. (2001). *Writing S-Functions*. The Mathworks, Inc.

Torres, S., J.A. Mendez, L. Acosta, M. Sigut, G.N. Marichal and L. Moreno (2001). A predictive control algorithm with interpolation for a robot manipulator with constraints. In: *Proc. 2001 IEEE Int. Conference on Control Applications*. Mexico City, Mexico. pp. 536 – 541.