Constrained generalized predictive control with dynamic programming

A. P. de Madrid, M. Santos, S. Dormido, F. Morilla

Abstract

This paper deals with a Predictive Control application involving Dynamic Programming. More specifically, how to solve the constrained optimization problems associated with predictive controllers in a straightforward and natural way, using Dynamic Programming. Some methods to reduce the computational load are introduced. Simulation results show that this technique and G.P.C. controllers produce similar system behavior.

Key words

Constrained Predictive Control, Dynamic Programming, Optimal Control, Mathematical Techniques.

1 Introduction

Model Based Predictive Control (M.B.P.C.) has been increasing its acceptance during the last years, substantially because some technologies such as modellization and identification, that permit the work in adverse environments, and digital computers, that by their speed and reliability permit to solve complex algorithms in real time, have reached a high degree of development.

Though most of the problems of real control can be solved with a simple PI controller, there exist some that require more advanced control techniques. In this case, Model Based Predictive Control offers a good relationship between performance and simplicity, and is an efficient and easy-to-use by the control engineers tool.

One of the reasons of its popularity in the industrial world is the possibility of explicitly incorporating constraints in the process, that appear in control problems due to so various reasons as physical limitations of the equipment, quality standards of the product, safety of the personal or equipment, environmental regulations, etc. Although the analytical resolution of the optimization problems in Predictive Control is easy in the non-constrained cases, their incorporation origin a big computational load; so it can be difficult to incorporate these techniques in the industrial world.

On the other hand, Dynamic Programming, developed by R. Bellman in the 50's decade (Bellman, 1957), is a very powerful computational tool for the resolution of dynamic optimization problems, whose use in practical control problems has not been widely extended by its requirements; however, with the current new technologies and computation elements development, these needs can be covered in large part without an excessive cost.

The application of Dynamic Programming to Predictive Control presents several advantages:

a) It is a technique opened to any type of system, defined even by a non-linear, non-continuous dynamics, etc.

b) The same method can be applied to any criterion to be minimized or maximized. That is, it admits any cost function, in a way such that it can be applied to all the model families used in Model Based Predictive Control: G.P.C., DMC, etc.

c) Finally, one of the most important characteristic because of its application to Predictive Control, is the possibility of incorporating constraints.

This work is structured of the following form. In section 2 the basis and the algorithm of Dynamic Programming are introduced, outlining some of its advantages. Section 3 summarizes some characteristic of Model Based Predictive Control. In section 4 it is shown how to apply Dynamic Programming to Constrained Predictive Control. Some application examples of this method are shown in section 5. Finally, some conclusions are briefly discussed in section 6.

2 Dynamic programming

2.1 Introduction to dynamic programming

Dynamic Programming, developed by R. Bellman (Bellman, 1957), provides an approach for solving optimization problems under very general conditions. In essence, this approach converts the simultaneous determination of the entire optimal control sequence into a sequential determination. The conversion is accomplished by using Bellman´s Principle of Optimality. The result is an iterative functional equation that can be solved by a digital computer retaining the desirable properties of the enumeration procedure -- such as constraint-handling capability and the computation of an absolute minimum -- while requiring much less computational effort.

Bellman states the Principle of Optimality in the following form:

"An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision."

Essentially, it tells us that an optimum decision policy has the property that any portion of an optimal trajectory from an intermediate state to the final state is itself the optimal trajectory from the intermediate state.

This allows us to determine a total optimum decision policy and the corresponding minimum cost function by starting at the end of the process and working backward one step at a time, considering only the decision at that stage. In determining this decision, we must consider both short-term cost at that stage and the long-term consequences of having to follow the optimal policy from the next state to which this decision takes us.

After having made this sweep backward through the stages, we can determine the optimum decision sequence and optimal trajectory in state space for any initial state by sweeping forward with the system equations, using the optimum decision policy at each stage to determine the next decision.

Problem formulation:

To apply Dynamic Programming, the problem must be formulated in the following form:

a) *The system equations*:

$$\underline{x}(k+1) = \underline{f}(\underline{x}(k), \underline{u}(k), k) \tag{1}$$

where $\underline{x}(k)$ is the n-dimensional state vector, $\underline{u}(k)$ the m-dimensional control vector, and k the stage variable.

b) *The performance criterion*:

$$J = \sum_{k=0}^{N} L(\underline{x}(k), \underline{u}(k), k) \tag{2}$$

This criterion provides an evaluation of a given decision sequence $\{\underline{u}(0), \underline{u}(1), \ldots, \underline{u}(N)\}$. This criterion is either an objective function to minimize or maximize, or a cost function to be minimized. Dynamic Programming can be applied to any criterion function having the Markovian property.

c) *Constraints*:

The constraints restrict on the values that the state variables and control variables can take. The state vector at stage k is constrained to be in the set $X(k)$ (fig. 1). The control vector applied at state $\underline{x}$, stage k, is constrained to be in the set $U(\underline{x},k)$. These constraints are expressed mathematically as:

$$\begin{aligned} &\underline{x} \in X(k) \subset R^n \\ &\underline{u} \in U(\underline{x},k) \subset R^m \end{aligned} \tag{3}$$

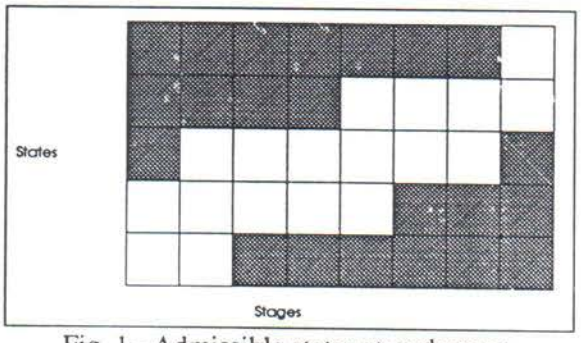

Fig. 1.- Admissible states at each stage.

Then the optimization problem can be stated as follows:

Given:

1) A system described by (1)

2) Constraints $\underline{x} \in X(k)$, $\underline{u} \in U(\underline{x}, k)$

3) An initial state $\underline{x}(0)$

Find:

The control sequence $\{\underline{u}(0), \underline{u}(1), ..., \underline{u}(N)\}$ that minimizes J in (2) while satisfying the constraints.

A problem will be solvable by Dynamic Programming if it verifies the Principle of Optimality. It can be assumed that f and L are continuous functions, and in base of the Weierstrass Theorem is assured the existence of a sequence that minimizes the function J.

2.2 Dynamic programming algorithm

First, state, control and stage variables are quantized. Constraints are used to restrict the admissible values of state and control variables.

The stage variable is indexed as k, $k = 0,1,...,N$. The set of quantized admissible states, X, is indexed as $\underline{x}_j$, j=1,2,..,J. The set of admissible controls, U, is indexed as $\underline{u}^{(m)}$, m = 1,2,...,M.

The *minimum cost function* $I(\underline{x}, k)$ expresses the minimum cost obtained by using any admissible control for the remainder of the process, beginning at any admissible state $\underline{x} \in X$ and at any stage k ($0 \le k \le$ N-1).

$$I(\underline{x},k) = \min_{\underline{u}\in U} \{L[\underline{x}(j),\underline{u}(j),k]\} \tag{4}$$

The algorithm has two steps:

First step: Iterative application of the functional equation to compute the minimum cost function $I(x, k)$:

$$I(\underline{x},k) = \min_{\underline{u}\in U} \{L[\underline{x},\underline{u},k] + I(\underline{f}(\underline{x},\underline{u},k),k+1)\} \tag{5}$$

```
for k = N-1 to 0                                   % for each stage from k = N-1 to k = 0
    for j = 1 to J                                 % for all x_j ∈ X at stage k
        for m = 1 to M                             % each of the admissible decisions u^(m) ∈ U are applied

            L_j^(m) = L [x_j, u^(m), k]            % cost at the current stage
            x_j^(m)(k+1) = f [x_j, u^(m), k]       % next state at stage k+1
            (x_j^(m), k+1) = P[ x_j^(m), N. I(x_j, k+1)]   % minimum cost at the states x_j^(m)(k+1) ∀ x_j ∈ X
                                                   (where P is an interpolation polynomial)
        end (admissible controls)
    end (admissible states)
end (stages)
```

A set of tables $I(x, k)$ is obtained, with the minimum cost for any quantized admissible state and its corresponding control at any stage k .

Second step: Calculation of optimal control

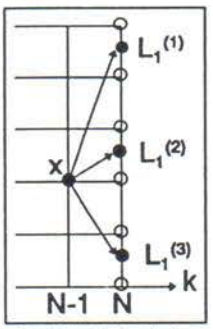

Fig. 2.- Optimal decision policy.

```
for k = N-1 to 0                                  % for each stage from k = N-1 to k = 0
    for j = 1 to J                                % for each quantized state x_j ∈ X
        J1(m) = L [x, u(m), k] + I (x(m) , k)     % cost at state x , stage k, and u(m)
        I(x, k) = min {J1(m)}
        û[x, k] = arg {min J1(m) }                % optimal control at stage k for each x
    end (admissible states)
end (stages)
```

The optimal control is determined as $\hat{u}[x, k] = u^{(m)}$ where m is the index of m for which (5) is minimized. The procedure consists in determining $I(x, k)$ and $\hat{u}[x, k]$ in terms of $I(x, k+1)$ for every quantized state x_j, j = 1,2,...,N beginning at stage (N-1) and continuing until stage 0 is reached.

The original problem was to find the optimal control sequence starting from the given initial state $x(0)$. This sequence can easily be determined from the Dynamic Programming solution. The first control in the sequence is evaluated as $\hat{u}[x(0), 0]$; the next state along the sequence is then found by applying the system difference equation to obtain $x(1)$. The procedure continues until the complete control sequence and the corresponding optimal trajectory have been obtained (fig. 3).

An important characteristic of this solution is that the optimal control is specified not just along this optimal trajectory, but for every admissible state at every stage.

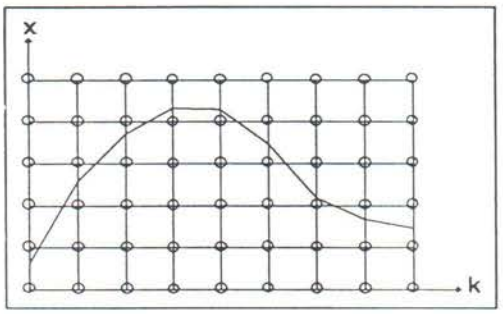

Fig. 3.- Recovery of an optimal trajectory.

If the cost function is convex and $U(x,k)$ is a convex set, then the minimum of the criterion is global. If the convex set is closed and the global minimum is in a finite point, then this point is in one of the extremes of the convex set of possible solutions. The number of extreme points is generally finite; in this case, this properly reduces considerably the search space

(Cooper and Cooper, 1981). Other possible reduction of the search space is obtained applying the admissible extreme values of the control in a given stage, that will bound the admissible trajectories in the state space and therefore will reduce the number of admissible states that are reachable (Dormido et al., 1970).

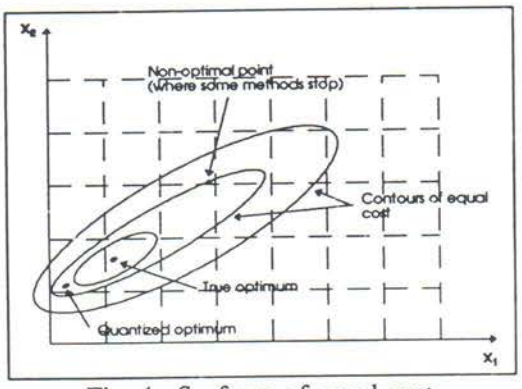

Fig. 4.- Surfaces of equal cost.

Dynamic Programming always determines an absolute minimum, not a relative minimum or a stationary point, because all the quantized admissible states are considered at each stage, and for all the admissible states all the possible quantized controls are evaluated. For that, within the precision of the quantification and the interpolation algorithm employed, a real global optimum is always obtained (fig. 4) (Larson and Casti, 1978; Larson and Korsak, 1970).

2.3 Advantages of dynamic programming

Briefly they can be summarized as follows:

1) One of the principal advantages of this method is the ability to handle system equations and performance criteria of a very general type (linear, nonlinear, discontinuous, non analytic, non quadratic, etc.). It is not necessary that they can be expressed in terms of well-known functions. All that is required is the existence of a rule for determining the values of the functions.

2) Simplicity to deal with constraints. They can be implemented as bounds of a very general type in a very simple way, because this numerical method avoids the complications inherent to the analytical techniques for the constrained problems. Furthermore, they diminish the set of states or control variables that are admissible. It is a very powerful tool to reduce the number of calculations to accomplish and analyze, reducing drastically the requirements of memory and calculation time.

3) Determination of an absolute optimum (not a relative minimum or maximum, or even worse, a stationary point), within quantization and interpolation precision.

4) Inherent simplicity, both in comprehension and implementation.

5) The feedback control or complete control policy solution. Furthermore, the solutions are applicable for an entire family of problems, not just for a single problem.

3 Introduction to model based predictive control

3.1 Model based predictive control

Under the name Model Based Predictive Control (M.B.P.C.), a diversity of control strategies families are grouped, that make explicit use of a model of the process to predict the value of the controlled variable during a time horizon (De Keyser, 1992).

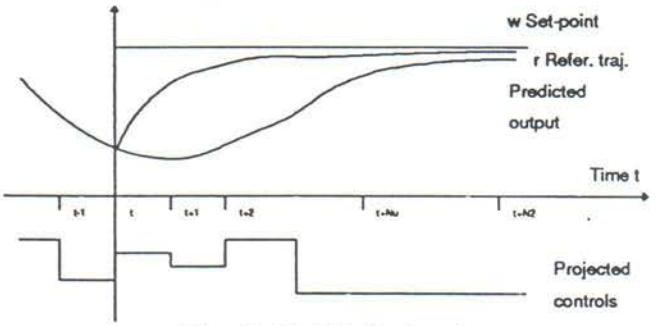

Fig. 5.- M.B.P.C. signals.

All these strategies present a series of common characteristics:

- At each instant t, the controlled output of the process $y(t+k)$ is predicted throughout a time horizon $k=1..N_2$. This is made by using a model of the process dynamics. The values $y(t+k/t)$ that are predicted depend on the future control signal $u(t+k)$, $k=0..N_2-1$.

- A reference trajectory $r(t+k)$, $k=1..N_2$ is defined for all the prediction horizon, that describes how to guide the process output from its current value $y(t)$ to the reference value w.

- The vector of future controls $\{u(t+k)\}$, $k=0..N_2-1$ is calculated, in such a way that it minimizes a certain function that depends on the predicted errors $\{r(t+k) - y(t+k)\}$, $k=1..N_2$. Then the first element $u(t)$ of the optimum control vector is applied to the real process.

- The prediction error between the measured process output and its prediction is used to make some type of reconstruction of the disturbances and / or adjustment of the model.

This procedure of four steps is repeated at each sampling instant. It is called a *receding horizon strategy.*

In general, the predictive controllers are able to solve a series of problems that are common to any adaptive control strategy:

- Control of non-minimum phase processes.

- Control of open loop unstable processes or with badly damped poles.

- Control of processes with an unknown or variable delay time.

- Control of processes of unknown order, without taking any special cautions.

3.2 Generalized predictive control (G.P.C.)

The M.B.P.C. version that appears to have had the most acceptability is called Generalized Predictive Control (G.P.C.), derived by Clarke, Mohtadi and Tuffs (Clarke et al., 1987a, 1987b; Clarke and Mohtadi, 1989). Because of its control parameters, it is possible to obtain several previous control strategies from it. Furthermore, this method includes the integral action in the closed loop *in a natural way*, which is required to obtain offset free control in the presence of non-zero mean disturbances.

G.P.C. is a control algorithm valid for MIMO systems. It is based on an ARIMAX process model. For a SISO system, it is given by the following form:

$$A(q^{-1})y(t) = B(q^{-1})u(t-k) + \frac{C(q^{-1})}{\Delta}\xi(t) \tag{6}$$

A and B are *n-th* and *m-th* order polynomials in the delay operator q^{-1}, with $a_0=1$; k is the delay time of the plant, and it will be considered equal to 1, ξ *(t)* represents an uncorrelated random sequence, or it can be also interpreted as zero-mean white disturbance, and Δ is the differencing operator $(1-q^{-1})$. This operator produces an incremental control law. The polynomial C is usually 1 (if not, it can be included in A and B).

A cost function is defined to minimize the error between the reference trajectory and the system output, including penalties on the control increments:

$$J(u,t) = E\left\{ \sum_{j=N_1}^{N_2} [r(t+j) - y(t+j)]^2 + \lambda_j \sum_{j=1}^{N_u} [\Delta u(t+j-1)]^2 \right\} \tag{7}$$

with $\Delta u(t+j) = 0$ for $j = N_u \ldots N_2$, and $r(t)$ is the reference signal. The other parameters are:

- N_1: the minimum output horizon.

- N_2: the maximum output horizon.

- N_u: the control horizon. For $t \geq N_u$, $\Delta u(t+j) = 0$.

- λ_j: the control-weighting factor.

The control signal is obtained from the minimization of this cost function (as the control signal is calculated conditioned to the available data in the instant t and assuming a stochastic disturbance model, the expected value $E(.)$ is used). The optimal control vector, for a constant value of λ, is given by:

$$\tilde{u} = (G^T G + \lambda I)^{-1} G^T (r - f) \tag{8}$$

$\tilde{u} = [\Delta u(t), \ldots, \Delta u(t+N_u-1)]$ is the vector of the future control increments, G is a N_2 x N_u matrix, containing the impulse response parameters of the plant $B / A\Delta$, and f is a vector whose

components f_j are the values of $y(t+j)$ that do not depend on the future control actions, that is, the predicted values for the process output with the information that it is available at instant t.

At instant t only $\Delta u(t)$ is calculated. This method is repeated at $t+1$ with the new values, to obtain the control sequence (receding horizon control strategy).

4 Constrained predictive control using dynamic programming

4.1 General problem formulation

To apply Dynamic Programming to a control problem, the following elements are needed:

- Equations describing the system dynamics: given by a predictive model of the process and its disturbances (De Keyser, 1992). The system outputs or the state variables will constitute the state vector, and the control signals will be the decision variables.

- Cost function to minimize: it will be a function of the predicted state (2) or error (9) (reference signal - predicted outputs) and the applied control signals.

$$J = \sum_{k=0}^{N_2} L(\underline{e}(k), \underline{u}(k), k) \tag{9}$$

Then the controller horizons and the rest of parameters will be fixed.

- System output constraints, or limits in the process state variables, according to the selected predictive model, as well as constraints in the control and its increments.

$$\begin{aligned} &\underline{x} \in X(k) \\ &\underline{u} \in U(\underline{x}, k) \\ &\Delta\underline{u} \in \Delta U(\underline{x}, k) \end{aligned} \tag{10}$$

- A method to compute the reference trajectory from the reference value.

The general Dynamic Programming procedure is applied:

1.- Minimal cost table I and optimum control table $Uopt$ are calculated using the final costs for all the possible final states, and then the usual method is applied with the imposed constraints.

Minimal cost function: $I(\underline{x}, k) = \min \{L[\underline{x}, \underline{u}, k]\}$ (11)

Iterative relationship: $I(\underline{x}, k) = \min \{L[\underline{x}, \underline{u}, k] + I(f(\underline{x}, \underline{u}, k), k+1)\}$ (12)

$\underline{x}$ values are obtained from the predictive model, applying the control signal $\underline{u}$.

The table will be calculated for all the prediction horizon N_2. If the control horizon N_u is smaller than N_2, the control signal will be constant from the stage N_u-1 to

the stage N_2-1 (see fig. 5). To do so, we must determine the optimum control signal to apply at each permitted state in the stage N_u-*1*. From each one, all the possible control signals will be tested, calculating the trajectory to the stage N_2 with constant control, as well as the cost of this trajectory. For each given departure state, the control signal that generates the smallest cost trajectory will be the optimum control signal to be applied in that state, and the state cost is the trajectory cost.

Once the optimum controls from the stage N_u-*1* have been obtained, the conventional Dynamic Programming algorithm is applied, with initial stage N_u-*1*, and its initial calculated cost.

2.- The optimum control sequence from the initial state is obtained for all the prediction horizon, but only the first value *u(t)* is applied (receding horizon strategy).

3.- The new value *y(t+1)* resulting from the application of the control signal is obtained, and it is used to apply this general method to the next stage (but if neither the reference signal nor the system model have been modified, tables are still valid).

This method can result quite laborious to compute, since every time that the reference signal or the system model vary all tables have to be calculated again. Thus, real time control of systems with fast dynamics could be difficult.

4.2 Practical aspects

There exist some solutions to reduce the calculation requirements:

- Take an averaged value for the reference signal *r*, assuming small deviations from this given value. As the system output will tend to the averaged value of *r* for which tables were computed (with the precision imposed by the partitions and interpolations), not to the real value *r(t)*, tables must be calculated *to make the error 0*. But this approximation can cause stationary error in the closed loop.

- Calculate several tables for different values of the control signal in advance. These tables will be used to interpolate the control signal for a non-computed reference. As this is a receding horizon strategy only the control for the first stage has to be interpolated (see fig. 6).

- If the constraints define a convex set and the cost function is also convex, then it is possible to use the properties of the convex sets and functions to reduce the number of admissible states and controls to explore.

Concerning the partitions in the state space, they are usually linear. This implies to work with the same resolution independently of how far from the desired value the signal is. It can be convenient that the partition is finer in the proximity of the desired value, and bulk far from it. The same reasoning could be applied to the quantification of the control signal; it improves the system performance if it is centered in the value needed to make the output equal to the reference. Logarithmic partition is shown in figure 7, but any other could be used.

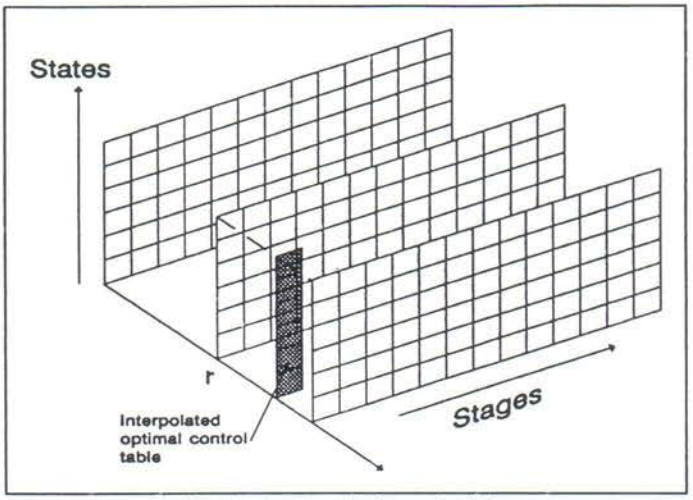

Fig. 6.- Interpolation of tables.

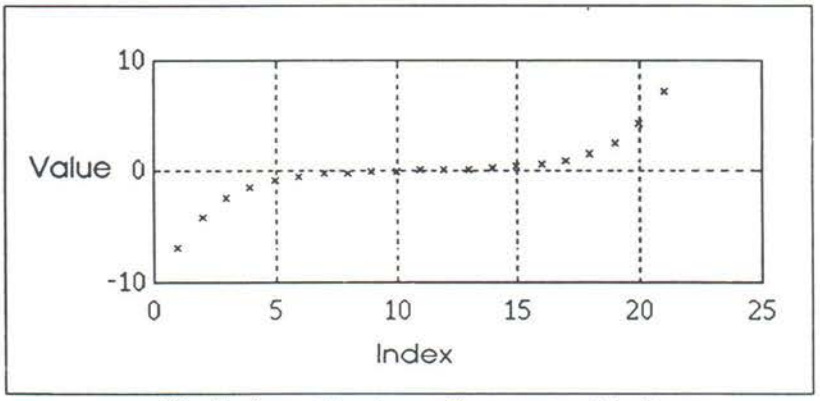

Fig. 7.- Logarithmic partition centered in 0.

5 Application examples

To test the proposed method, consider a first-order linear system with a pure dead time of one sampling period, unity gain and time constant 1s; the sampling period T is 0.1s. It is described by the following polynomials:

$$A = 1 - 0.9048q^{-1}$$
$$B = 0.0952q^{-1} \qquad (13)$$

Selected controller parameters are N_1=1, N_u=1, N_2=10, λ=0. The control signals are constrained by 0 and 5, and the output signal is also constrained by 0 and 4; with these parameters, and for the reference signal used, the constraints will not be make active, so this case is similar to the non-constrained one. If the optimal control tables are computed with a linear interpolation algorithm, the controller performance is as shown in figure 8. The system response is good and rapid, similar to the non-constrained "classic" G.P.C.'s response (fig. 9), but the control signal is no so smooth, because of the precision of partition used (for this case, with increments of 0.5 units), so a smaller partition would produce a smoother control signal.

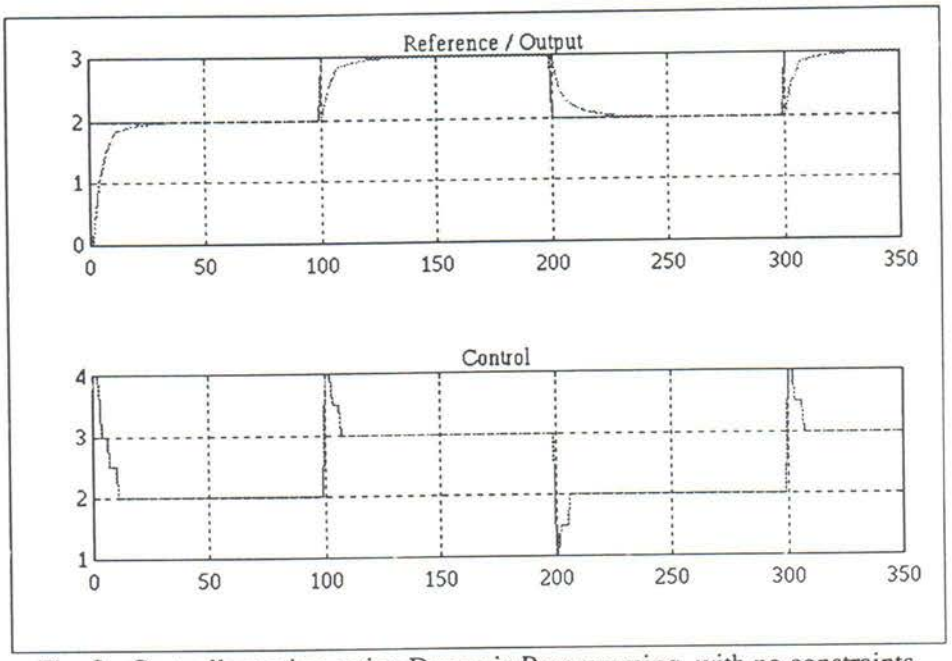

Fig. 8.- Controller action, using Dynamic Programming, with no constraints.

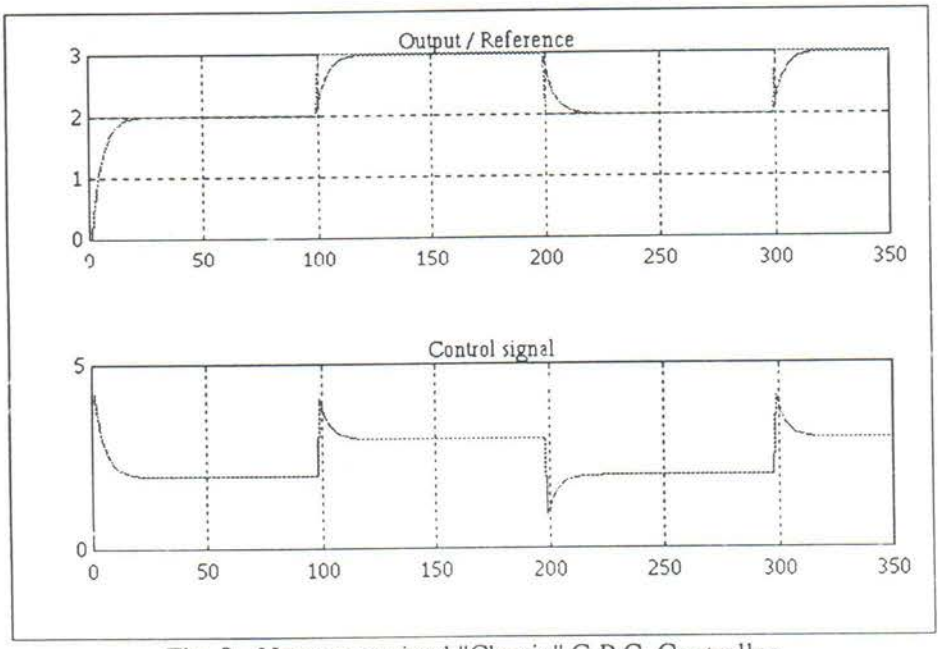

Fig. 9.- Non-constrained "Classic" G.P.C. Controller.

To show how constraints activate, consider a maximum control value of 3, with steps of 0.5 again, and no constraints in the control increments. This case is illustrated in figure 10. System output is good, but a bit slower due to the control saturation.

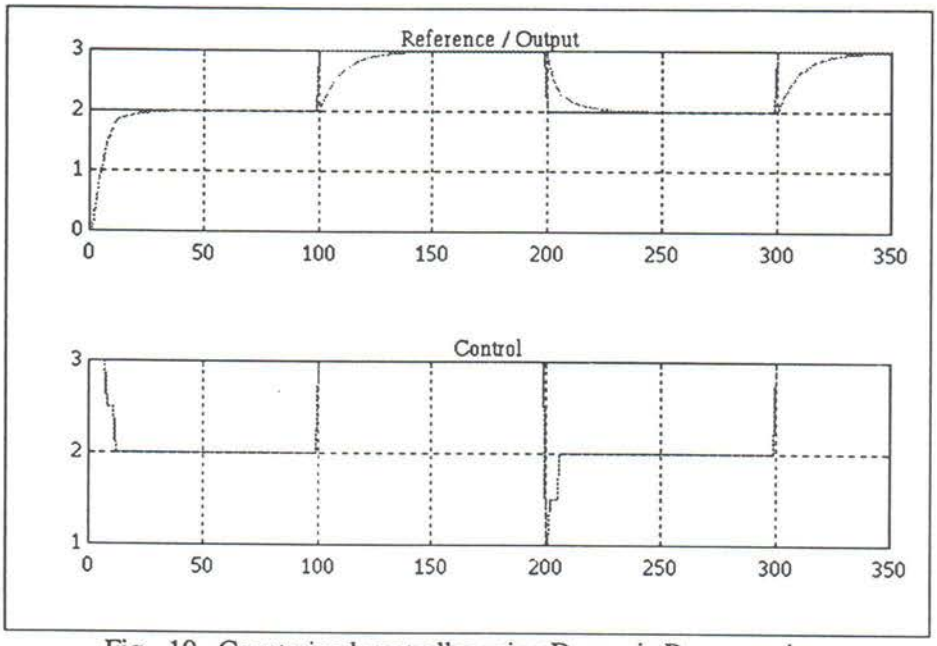

Fig.- 10. Constrained controller using Dynamic Programming.

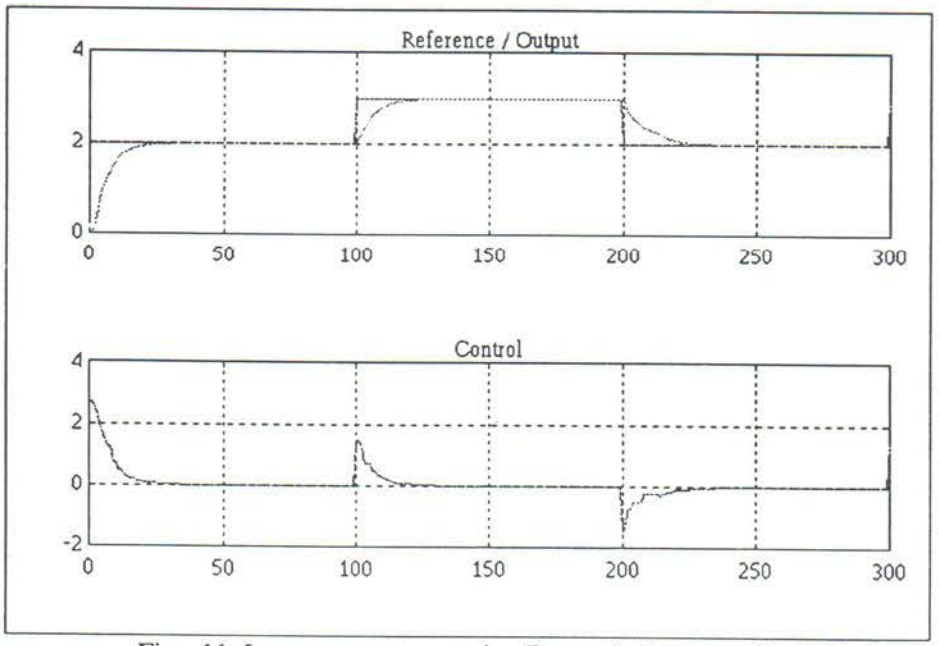

Fig.- 11. Integrator system, using Dynamic Programming

Dynamic Programming produces a quantized optimal solution, but it is accurate enough to control a more complicated system. Let us consider now the case of a single integrator, defined by the continuos transfer function G(s)= 1 / s. For this case, we have used two sets of optimal control tables: the first one is used when the absolute value of the error is greater than 0.25, and the second one, with a higher precision, when it is less or equal to 0.25.

These two degrees of precision make possible to get very accurate solution, as we can see in figure 11, but reducing the number of states to consider.

6 Conclusions

In this work, it has been shown how to apply Dynamic Programming to Predictive Control. It has proved to be a simple and powerful tool to solve the constrained optimization problems associated with these controllers. The computational load can be drastically reduced, using pre-computed optimal control tables and some other methods that have been described. Some application examples have shown that, as Dynamic Programming provides a global optimum solution (within the accuracy imposed by the partitions and interpolations), system performance is comparable to the results obtained by a "classic" G.P.C. But this is a more general method, able to handle non-linear models and constraints and non-quadratic cost functions, without any additional effort.

Further work includes the study of control algorithms based on the Forward Dynamic Programming formulation (Larson and Casti, 1978), which could be a more intuitive way to deal with the problem, the study of how partitions affect the system dynamics, stability, etc., and the straightforward development of new types of predictive controllers based on non-quadratic cost functions and non-linear system models.

7 Acknowledgments

The authors wish to acknowledge the economical support of the Spanish CICYT under grant TAP 122/92.

8 References

Bellman, R. (1957). *Dynamic Programming*. Princeton University Press, N.J.

Bitmead, R. B., Gevers, M. and Wertz, V. (1990). *Adaptive Optimal Control: The Thinking Man's GPC*. Prentice Hall International. Series in Systems and Control Engineering.

Camacho, E. F. (1993). *Constrained Generalized Predictive Control*. IEEE Transactions on Automatic Control, Vol 38, No. 2, February, pp. 327-332.

Clarke, D. W., Mohtadi, C. and Tuffs, P. S. (1987a). *Generalized Predictive Control. Part I. The Basic Algorithm*. Automatica, Vol. 23, No. 2, pp. 137-148.

Clarke, D. W., Mohtadi, C. and Tuffs, P. S. (1987b). *Generalized Predictive Control. Part II. Extensions and Interpretations*. Automatica, Vol. 23, No. 2, pp. 149-160.

Clarke, D. W., Mohtadi, C. (1989). *Properties of Generalized Predictive Control*. Automatica, Vol. 25, No. 6, pp. 859-875.

Clarke, D. W., Scattolini, R. (1991). *Constrained Receding-Horizon Predictive Control.* IEE Proceedings-D, Vol. 138, No. 4, July, pp. 347-354.

Cooper, L. and Cooper, M. W. (1981). *Introduction to Dynamic Programming.* Pergamon Press.

De Keyser, R. M. C. (1992). *The MBPC Methodology.* Intensive Training Course on Model Based Predictive Control. U.N.E.D., Madrid.

Demircioglu, H., Clarke, D. W. (1992). *CGPC with Guaranteed Stability Properties.* IEE Proceedings-D, Vol. 139, No. 4, July, pp. 371-380.

Dormido, S. et al. (1970). *Consideraciones Sobre la Regulación de Sistemas Mediante Técnicas de Programación Dinámica.* Revista de Automática, No. 10, pp. 29-34.

Lamber, E. P. (1987). *Process Control Applications of Long-Range Prediction.* Report No OUEL 1715/87. Department of Engineering Science. University of Oxford.

Larson, R. E. and Casti, J. L. (1978). *Principles of Dynamic Programming.* Ed. Dekker, USA.

Larson, R. E. and Korsak, A. J. (1970). *A Dynamic Programming Succesive Approximations Technique with Convergence Proofs. Part II: Convergence Proofs.* Automatica, Vol 6, pp. 253-260, Pergamon Press.

Tsang, T. T. C., Clarke, D. W. (1988). *Generalized Predictive Control with Input Constraints.* IEE Proceedings-D, Vol. 135, No. 6, November, pp. 451-460.