Real-Time Predictive Control of a Servo Engine

Martin Herceg, Michal Kvasnica, Ľuboš Čirka, Miroslav Fikar

Abstract
This paper deals with real-time implementation of model predictive control techniques to a laboratory servo engine. To cope with the nonlinear behavior of the plant, the piecewise affine (PWA) modelling framework is adopted. It is shown that PWA models are especially well suited to describe systems with deadzone-type of nonlinearities. Subsequently we show two approaches to design an MPC controller. First scheme is based on solving an optimization problem of finite size, while the second approach yields a control law which drives the states of the plant to respective references in minimum time. To address the issue of real-time applicability of the obtained controllers we propose to use the concept of parametric programming.

Key words: model predictive control, minimum time control, deadzone

Introduction

Control of systems which express nonlinear behavior is a long standing problem both in control theory and practice. Several approaches can be adopted to design control laws for such systems, ranging from application of neural networks (see e.g. [12]), through adaptive control techniques [15], up to classical PID/LQ control theory where the nonlinearities are approximated using linearization techniques [14]. The problem gets more complicated if certain process constraints, such as limits on manipulated or output variables, have to be respected by the control policy. It is without doubt that Model Predictive Control (MPC) [11] is one of the most efficient control techniques when constraint satisfaction and optimal performance is required. The MPC approach, in its most general form, allows to use almost arbitrary prediction models, including nonlinear [1], [9] or hybrid [2] models. Contrary to classical PID or LQR controllers, the MPC strategy naturally incorporates the process constraints such that the manipulated inputs generated by the scheme never violate safety limits.

MPC is an optimization-based strategy in which the values of the manipulated variables are decided such that certain performance criterion is minimized (or maximized). This allows one to incorporate e.g. economical aspects directly into the control design. One drawback of the MPC approach is that the optimization problem has to be solved repetitively at each sampling instance for a given value of the measured initial condition. Size and complexity of the optimization therefore significantly limit the minimal admissible sampling time of the plant. To address this issue and to allow real-time implementation of MPC to processes with sampling times in order of milli- and micro-seconds, the concept of parametric solutions to MPC problems has been proposed by [3]. In this approach the optimization problem is solved for all possible initial conditions which satisfy given constraints. The solution takes a form of a look-up table, which can be evaluated in real-time using a simple set-membership test. The test can be implemented[1] as a sequence of matrix multiplications and logical comparisons

Even though the concept of MPC can be coupled with a nonlinear model of the plant dynamics, the resulting nonlinear optimization problem is often very difficult to solve to a global minimum in admissible time. Therefore the concept of hybrid systems [2] has been proposed to derive simpler models which approximate the nonlinearities with sufficient precision. One possible way is to perform multiple linearizations around different operating points. Each such linearized model is then valid in a domain whose boundaries are linear. Using simple rules from propositional logic it can be shown that the model can be described mathematically by a set of inequalities involving continuous and discrete (boolean) variables. A particular mathematical representation, commonly referred to as the piecewise affine (PWA) modelling framework, is discussed in this paper.

In the sequel we show how to derive a suitable PWA model of a plant which involves deadzone-type of nonlinearities. Specifically, we investigate a real servo engine device, which represents a mechanism for operating valves in pipes. As will be seen later, the PWA model describes the real behavior of the plant with a great precision.

Once the PWA model of the plant is available, we propose to solve the MPC problem parametrically in order to obtain a feedback law which satisfies control design requirements, such as constraint satisfaction or certain performance criteria. One of them being a need for a fast (and ideally the fastest possible) transition from one steady-state to an another. First we investigate the case where an optimal control problem formulated over a finite prediction horizon is solved parametrically. However such approach, due to its nature, cannot guarantee that the transition from one set-

[1] The implementation can be further simplified using advanced search strategies, such as binary search trees of [13].

point to an another will be performed as quickly as the constraints and the dynamics permit. Therefore we adopt the concept of minimum-time control [7]. In such scheme the states of the plant are forced to move to a prescribed terminal set around the origin in the least possible number of steps, while respecting constraints on input and state variables. There is, however, little known about how to synthesize a minimum-time controller for problems other than regulation towards the origin. The main contribution of this paper is therefore an extension of the minimum-time principle to cases where the reference signal is time-varying.

Description of the Physical Setup

The laboratory servo engine is a mechanical device which represents a valve opening mechanism. It consist of a rotational flywheel attached to an electric engine. The manipulated variable is a voltage ranging from -15V to +15V generated by an actuator, which drives the rotation of the flywheel counterclockwise (when negative voltage is applied) or in the opposite direction (positive voltage). The angular velocity of the flywheel is the output signal which can be measured. The flywheel is connected to a circular pointer which indicates the position of the valve. The position of the wheel is the secondary output which ranges from the position “fully open” (0 degrees) to “fully closed” (360 degrees). A front view of the servo engine is illustrated in Fig. 1. Notice the two circular pointers with red triangles in the foreground of the two discs. The left one is the position indicator and the right one represents the setpoint which can be manipulated manually. Under the left pointer there is a magnetic brake whose force can be adjusted by turning the red colored switch. The brake is considered to be an external disturbance.

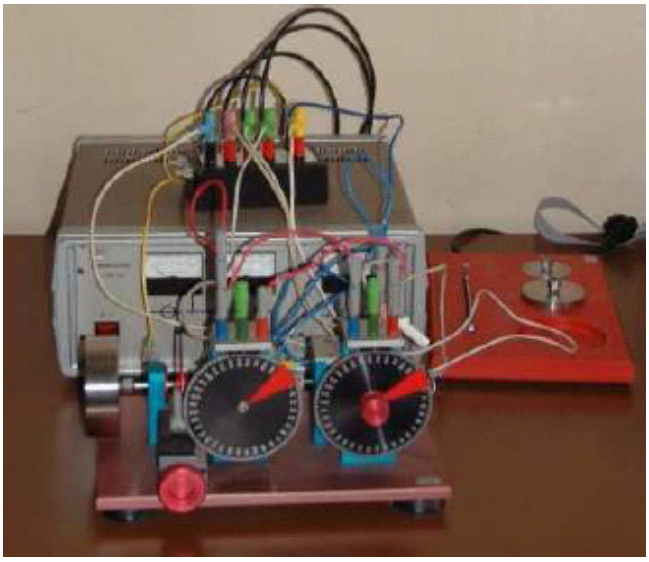

Fig. 1 Front view of the laboratory servo engine.

The laboratory servo engine is connected to a personal computer via three devices: an actuator, a transducer and the connector CP1102. The devices actually convert the operating range of the dSPACE input-output card (-10V, +10V) to the actuator voltage range (-15V, +15V). MATLAB’s Real Time Workshop (RTW) serves as a tool for implementing the control policy.

Hybrid Model of the Servo Engine

Theoretical background

The mathematical model of the servo engine originates from the second Kirchhoff’s law of preservation of the current in a knot and from the inertia conservation law. The voltage generated in the servo engine u is given by

$$u = Ri + L\frac{\mathrm{d}i}{\mathrm{d}t} + U_i \tag{1}$$

where i is the current in the solenoid, R denotes resistance, L inductance and U_i is an inducted voltage. The inducted voltage U_i depends linearly on the velocity of the flywheel, i.e. $U_i = C\omega$ where C is a constant and ω denotes the angular velocity of the flywheel. The generated torque M is given by

$$M = M_s + J\frac{\mathrm{d}\omega}{\mathrm{d}t} \tag{2}$$

with M_s denoting the load needed at start and J denotes the inertia of the flywheel. Using the linear relation between the torque and current $M = Ci$ we end up with a system model described by following two equations:

$$\frac{\mathrm{d}i}{\mathrm{d}t} = -\frac{R}{L}i - \frac{C}{L}\omega + \frac{1}{L}u \tag{3}$$

$$\frac{\mathrm{d}\omega}{\mathrm{d}t} = \frac{C}{J}i - \frac{M_s}{J} \tag{4}$$

By plugging (4) into (3), the model can be transformed into an ordinary linear differential equation between the input voltage u and the measured output ω as

$$LJ\ddot{\omega} + \frac{RJ}{C^2}\dot{\omega} + \omega = \frac{1}{C}u - \frac{RM_s}{C^2} \tag{5}$$

where $\dot{\omega}$ denotes the time derivative of the angular velocity.

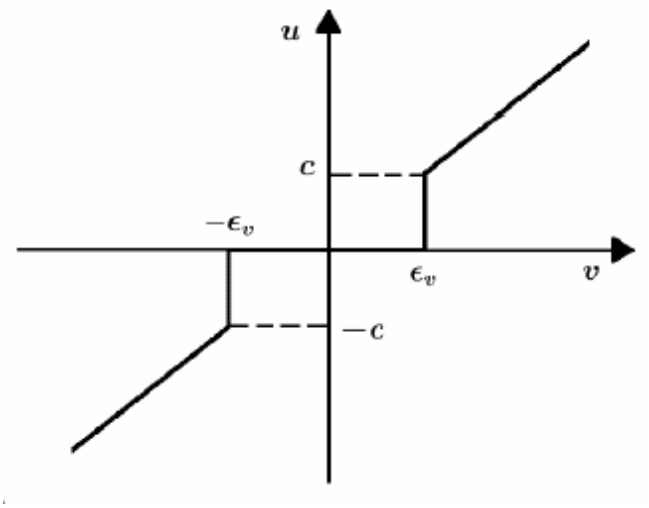

Fig. 2 Graphical illustration of a deadzone in the servo engine

The constant term on the right hand side of (5) contains an unknown parameter M_s which only influences the dynamics if the speed is zero. It represents the initial load of the solenoid and can be neglected when the flywheel cruises at a nonzero speed. This phenomenon is known in process

control as a nonlinear deadzone behavior. Without loss of generality we shall rewrite the equation (5) to a general form

$$T_1 T_2 \ddot{\omega} + (T_1 + T_2)\dot{\omega} + \omega = Ku \tag{6}$$

with time constants T_1, T_2 and a gain K. In the sequel we will denote by v the measurement (in volts) of the angular velocity ω.

The actually applied input is now denoted as u_p and its relation with the voltage u is modelled as given in [5] by

$$u_p = \begin{cases} m_d(u-c)+c & \text{if } v > \varepsilon_v \\ 0 & \text{if } -\varepsilon_v \le v \le \varepsilon_v \\ m_d(u+c)-c & \text{if } v < -\varepsilon_v \end{cases} \tag{7}$$

where m_d is the slope, and c and ε_v are the ranges of the deadzone. The dependence (7) is graphically depicted in Fig. 2.

Piecewise Affine Models

PWA systems represent a powerful modelling framework to describe the behavior of nonlinear systems by multiple linearizations at different operating points. Such description can be efficiently integrated into model predictive control strategies by introducing logical variables [2]. In the sequel a brief overview of a PWA model will be given.

Consider the class of linear hybrid systems which can take the discretized form

$$\begin{aligned} x_{k+1} &= f_{PWA}(x_k, u_k) \\ &= A_i x_k + B_i u_k + f_i \quad \text{if} \quad \begin{bmatrix} x_k \\ u_k \end{bmatrix} \in D_i \end{aligned} \tag{8}$$

where $x_k \in \mathbf{R}^n$ is the system state, $u_k \in \mathbf{R}^m$ is the manipulated input, and k denotes a sampling instant. Furthermore, $D = \bigcup_{i=1}^{n_D} D_i$ denotes the domain of f_{PWA}, which is a nonempty compact set consisting of a finite number of convex polytopes in the joint $x-u$ space. Formally D_i can be defined by

$$D_i := \left\{ \begin{bmatrix} x_k \\ u_k \end{bmatrix} \in \mathbf{R}^{n+m} \mid G_i^x x_k + G_i^u u_k \le G_i^0 \right\} \tag{9}$$

where G_i^x, G_i^u, G_i^0 are constant matrices of suitable dimensions specifying the borders of the ith region. Notice that state and input constraints can directly be incorporated into (9).

PWA Model of the Servo Engine

The motivation to use PWA systems to model the servo engine device stems from presence of the deadzone description, as given by (7). In the previous work of [8] the effect of the deadzone was neglected, since only initial conditions excluding zero starting velocity have been considered. However if one wants to design a control law which properly operates the plant even if it starts from the rest (i.e. $v_0 = 0$), an appropriate model of the deadzone is of imminent importance.

The deadzone description (7) can be straightforwardly modelled as a PWA system of the form (8) with three modes:

$$x_{k+1} = \begin{cases} Ax_k + Bu_k & \text{if } |v| \ge \varepsilon_v \\ Ax_k + Bu_k & \text{if } |v| < \varepsilon_v \text{ and } |u| \ge c \\ Ax_k & \text{if } |v| < \varepsilon_v \text{ and } |u| < c \end{cases}. \tag{10}$$

Here c and ε_v denote the breakpoints of the deadzone. The state vector $x_k = \begin{bmatrix} v_k & y_k \end{bmatrix}^T$ is composed of the angular velocity of the flywheel and its position, respectively. The numerical values of the matrices describing the state-space dynamics $x_{k+1} = Ax_k + Bu_k$ can be either obtained directly from (6), or derived from an identified model of the plant. The latter approach has been used by [8], where the system behavior has been identified as a first-order system

$$T\dot{v} + v = Ku \tag{11}$$

with the time constant $T = 6.7057\ \text{s}$ and the gain $K = 0.1271$. Values of A and B are then calculated by discretizing (11) using the sampling time $T_s = 0.7\ \text{s}$

Notice that the PWA description (10) is defined over a non-convex domain. In order to convert the model into a form where each mode is valid over a convex region, the non-convex domains $|v| \ge \varepsilon_v$ and $|u| \ge c$ each have to be split in two parts. The resulting PWA model is then given by following 5 modes:

$$x_{k+1} = \begin{cases} Ax_k + Bu_k & \text{if } v \ge \varepsilon_v \\ Ax_k + Bu_k & \text{if } v \le \varepsilon_v \\ Ax_k + Bu_k & \text{if } -\varepsilon_v \le v \le \varepsilon_v \ \wedge \ u \ge c \\ Ax_k + Bu_{kk} & \text{if } -\varepsilon_v \le v \le \varepsilon_v \ \wedge \ u \le -c \\ Ax_k & \text{if } -\varepsilon_v \le v \le \varepsilon_v \ \wedge \ -c \le u \le c \end{cases} \tag{12}$$

The manipulated voltage is supposed to be constrained by $-10 \le u \le 10$, while the angular velocity of the flywheel has to satisfy $-1.5 \le v \le 1.5$.

Model Validation

A test scenario was used to validate the PWA model (12) versus measured data. By imposing a known control profile, output from the model was compared with the data collected from the device. As can be seen from Figure 3, the model approximates the real behavior of the plant with sufficient accuracy. Successful modelling of the deadzone is clearly seen from the first 6 seconds of the comparison. Since voltage inputs ranging from -1V to +1V have been applied during that period, the deadzone forbids the flywheel to start moving. This is correctly captured by the model. Once the input voltage exceeds deadzone threshold, the model correctly switches to a different operating mode where the input starts to influence the angular velocity of the flywheel.

Model Predictive Control

In this section we propose two MPC strategies which can be used to control an electrical servo engine described in the previous sections. The first approach is based on optimizing the sequence of manipulated variables over a finite prediction horizon, while the second scheme is based on the so-called minimum-time principle. Although the minimum-time case has already been covered extensively e.g. by [7], the contribution of this section is a novel way of synthesis of minimum-time controllers which drive the system states to

prescribed time-varying reference trajectories other than the origin.

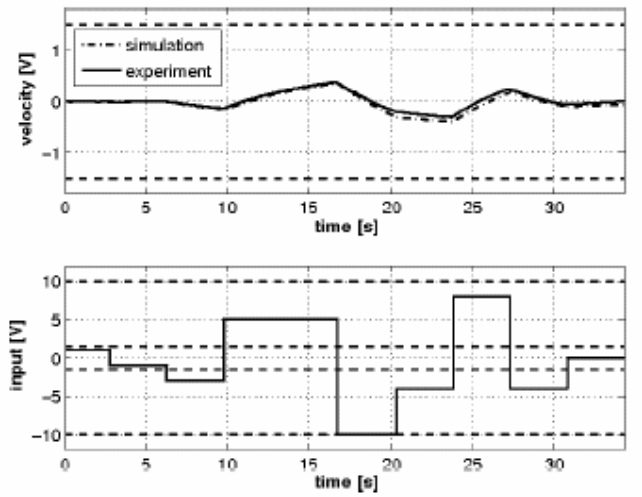

Fig. 3 Comparison between the model behavior and measured signals.

Constrained Finite Time Optimal Control

The Constrained Finite Time Optimal Control problem can be formulated as to find an admissible solution to

$$\begin{aligned} \min_{u_0,\ldots,u_{N-1}} \quad & \sum_{k=0}^{N-1} \left\| R(u_k - u_{ref}) \right\|_p + \left\| Q(x_k - x_{ref}) \right\|_p \\ \text{s.t.} \quad & x_{k+1} = f_{PWA}(x_k, u_k) \\ & x_k \in X \\ & u_k \in U \end{aligned} \tag{13}$$

where the index p denotes a linear norm (either $p=1$ or $p=\infty$). R and Q represent the weighting matrices which penalize, respectively, the deviation of the input and state signals from given references. We will assume $u_{ref}=0$ throughout the rest of the paper[2]. The PWA model of the controlled system is incorporated as the equality constrain in (13). The constraints X and U are assumed to be compact polyhedral sets containing x_{ref} and u_{ref} in their respective interiors.

If the initial state x_0 as well as the value of the reference signal x_{ref} are both known, the optimization problem (13) can be formulated and solved as a Mixed Integer Linear Program (MILP). The integer variables are used to model the different modes of the PWA model. However it is well known that MILP problems are, in general, NP-hard [6]. Even though efficient branch-and-bound algorithms exist, the complexity of the optimization problem can easily be prohibitive for real-time applications due to time constraints. Specifically, in order to apply the MPC principle to our servo engine plant, the optimization problem has to solved within the given sampling time of 0.7 seconds. To address this issue, [3] proposed to solve the optimization problem parametrically for all possible values of the initial condition x_0 which satisfy the constraints. The advantage of the parametric solutions is that the optimal control input can be obtained in real-time by simply evaluating a look-up table.

Theorem 1. *(Solution to CFTOC [4]) The solution to the optimal control problem (13) with* $p \in \{1, \infty\}$ *and a linear state-update in (13) is a time-varying piecewise affine state feedback control law of the form*

$$u^*(x_k) = F_i^k x_k + G_i^k \quad \text{if} \quad x_k \in P_i^k \tag{14}$$

and the optimal value function is a time-varying piecewise affine function of the state

$$J_k^*(x_k) = \Phi_i^k x_k + \Gamma_i^k \quad \text{if} \quad x_k \in P_i^k \tag{15}$$

where $P_i^k = \left\{ x \in R^n \mid P_i^k x \le P_i^0 \right\}$ *is a polyhedral partition of the set* X^k *of feasible states* x_k *at time* k *with* $k = 0,\ldots,N-1$.

The case where the state-update equation in (13) is driven by a PWA model of the form (12) has also been handled by [4] where it is shown that the parametric solution to (13) also takes a form of a PWA feedback law (14) defined over a (possibly non-convex) polyhedral domain. Theorem 1 therefore provides a powerful result as it suggests that MPC problems can be solved parametrically and implemented in real-time by a simple table look-up. The table is parameterized by the x_0 parameter.

However if the reference signal x_{ref} is wanted to be time-varying, one has to extend the vector of parameters to include this quantity. This can be done easily by augmenting the state vector as

$$\tilde{x}_k := \begin{bmatrix} x_k \\ x_{ref} \end{bmatrix} \tag{16}$$

and changing the penalty matrix Q to penalize the difference $x_k - x_{ref}$, i.e.

$$\tilde{Q} = \begin{bmatrix} Q & -Q \end{bmatrix} \tag{17}$$

With this augmentation the optimal control problem (4.1) can be rewritten as

$$\begin{aligned} \min_{u_0,\ldots,u_{N-1}} \quad & \sum_{k=0}^{N-1} \left\| R u_k \right\|_p + \left\| \tilde{Q}\tilde{x}_k \right\|_p \\ \text{s.t.} \quad & \tilde{x}_{k+1} = \tilde{f}_{PWA}(\tilde{x}_k, u_k) \\ & \tilde{x}_k \in \tilde{X} \\ & u_k \in U \end{aligned} \tag{18}$$

with

$$\begin{aligned} \tilde{x}_{k+1} &= \begin{pmatrix} A_i & 0 \\ 0 & I \end{pmatrix} \tilde{x}_k + \begin{pmatrix} B_i \\ 0 \end{pmatrix} u_k \\ &= \tilde{f}_{PWA}(\tilde{x}_k, u_k) \end{aligned} \tag{19}$$

To sum up, the parametric solution to the CFTOC problem (18) is a look-up table parameterized by the initial condition x_0 and the reference signal x_{ref}. The table can be calculated efficiently using e.g. the Multi-Parametric Toolbox [10]. Performance of the MPC scheme can be tuned by appropriately adjusting the weighting matrices Q and R, and by a suitable choice of the prediction horizon N. However there is no a-priori guarantee that the control policy

[2]This is a reasonable assumption for position control if the model description already includes an integrator. If it were not the case, one could optimize over $\Delta u_k = u_{k+1} - u_k$ instead of u_k in order to eliminate the steady-state offset, and use $\Delta u_{ref} = 0$ to indicate that the manipulated variable should be kept constant once the states converged to the reference.

will drive the state of the plant to the reference signal as quickly as possible. To achieve such goal, we use the concept of minimum-time control.

Minimum-time Control

The task of designing a control law which drives system states to a desired target set in minimum-time has been addressed by [7]. The authors propose an algorithm which first designs a control invariant set around the origin and subsequently iteratively constructs the control law which steers the system to such set in a minimal number of steps. A drawback of such approach stems from the fact that it can only be used for regulation problems, i.e. for tasks of driving the system states to the origin. However in applications like the one considered here, we require a subset of states to converge to non-zero time-varying reference signals.

Therefore we propose an alternative way of computing the initial target set, while the rest of the algorithm remains unchanged. The target set has to be designed such that once states x are contained in such set, there is a control law which drives them to the reference x_{ref} in a finite number of steps. Moreover, the target set has to be designed such that it satisfies the set invariance property.

Definition 1. *(Control Invariant Set) A control invariant set ψ is the set of states for which there exist control inputs $u_k \in U$ such that if $x_0 \in \psi$, then all subsequent state updates $x_{k+1} = f_{PWA}(x_k, u_k)$ also belong to the set ψ for $k = 0,\ldots,\infty$.*

To find a suitable target set and a control law active in such set which brings system states to desired trajectories in a fixed number of steps, we suggest to solve the following feasibility problem:

$$\begin{aligned} \text{find} \quad & u_k = f(\tilde{x}_k) \\ \text{s.t.} \quad & x_{k+T} = x_{ref} \\ & \tilde{x}_{k+1} = \tilde{f}_{PWA}(\tilde{x}_k, u_k) \\ & \tilde{x}_k \in \tilde{X} \\ & u_k \in U \end{aligned} \tag{20}$$

Notice that (20) is in fact a constraint on $\tilde{x}_{k+T}$, since it can be written as $\begin{bmatrix} I & -I \end{bmatrix}\tilde{x}_{k+T} = 0$ with $\tilde{x}_{k+T} = \begin{bmatrix} x_{k+T} & x_{ref} \end{bmatrix}^T$. If the problem (20) is solved parametrically, according to Theorem 1 one obtains the feasible set Ω of parameters $\tilde{x}_0$ which satisfy given constraints. The set takes a form of a (possibly non-convex) union of finite number of convex polytopes. In addition, the control law $u = f(\tilde{x})$ will take a form of the PWA state-feedback as in (14). The equality constraint (20) assures that the control law will be in a form such that once the state resides in the set Ω, it will be steered to the respective reference in, at most, T steps. Therefore the obtained control law can be viewed as a dead-beat controller for the class of hybrid systems. Notice that, ideally, one would choose $T = 1$ in (20). However, if the process dynamics (20) includes transport delays or unstable zeros, it is necessary to choose $T > 1$ in order for (20) to have a feasible, non-empty solution.

Also notice that the set Ω of the states for which (20) is feasible satisfies the control invariance property by construction. This is due to the fact that every state which starts from Ω is forced to remain in the set by the equality constraint (20).

Once the control invariant set Ω is available, it can be used to design a minimum-time controller. The control law is constructed iteratively, where at each step the following optimization problem for prediction horizon $N = 1$ is solved parametrically:

$$\begin{aligned} \min_{u_k} \quad & \left\| Ru_k \right\|_p + \left\| \tilde{Q}\tilde{x}_k \right\|_p \\ \text{s.t.} \quad & \tilde{x}_{k+1} = \tilde{f}_{PWA}(\tilde{x}_k, u_k) \\ & \tilde{x}_{k+1} \in T_{set} \\ & \tilde{x}_k \in \tilde{X} \\ & u_k \in U \end{aligned} \tag{21}$$

with (21) representing a terminal set constraint. The minimum-time algorithm of [7] can now be stated as follows:

1. Solve the problem (20) parametrically and denote by Ω the set of states for which the problem is feasible.
2. Set the iteration counter $i = 0$ and set $S_i = \Omega$.
3. Set $T_{set} = S_i$ in (21) and solve (21) parametrically by considering $\tilde{x} = \begin{bmatrix} x & x_{ref} \end{bmatrix}^T$ as the parameter. Denote the feasible set of the problem (34) by S_{i+1}.
4. If $S_{i+1} = S_i$, stop, the algorithm has converged.
5. Increase i by 1 and jump back to Step 3.

At every run of Step 3 of the algorithm above, a control law of the form $u = F_i\tilde{x} + G_i$ is generated according to Theorem 1. Since the dynamics in (21) is hybrid, the feasible set $S_i = \bigcup_i P_i$ will be a non-convex union of finite number of convex polytopes. The set is then used at the next iteration as a new target set constraint. The algorithm terminates if at two subsequent iterations no new states have been added to S_i. Since one-step problems are solved at each step, the controller guarantees that all states will enter the initial terminal set Ω in the least possible number of steps. Subsequently, once the states arrive to the set Ω, the control laws which have been obtained as a solution to (20) will drive them to the respective reference in a dead-beat fashion.

Experimental Results

The servo engine is controlled via a dSPACE I/O card and the control algorithm is implemented using Matlab's Real Time Workshop. The feedback laws in the form of a look-up table are exported to C-code along with a policy which searches the table for a region which contains the actual state measurements. The parametric solutions have been obtained by the Multi-Parametric Toolbox [10], which is also able to create a respective C-code representation of the controller logic and data in a user-friendly way. The optimal control policy as well as the minimum-time controller of have been investigated in real-time experiments.

CFTOC

The controller was synthesized with symmetric unit weights $Q = 100I$, $R = I$, and the prediction horizon set to $N = 4$. Using the MPT toolbox, a parametric solution consisting of 602 polytopic regions in the 3-dimensional state-space (velocity, position, and the reference) have been obtained. The solution was subsequently exported to C-code, implemented

using the RTW framework, and experimental data have been measured. Figure 4 shows the performance of the CFTOC controller when tracking a given position reference. Despite the presence of the deadzone nonlinearity, the controller acts appropriately even when the flywheel is at standstill. This shows the benefits of the MPC approach since it allows to include the description of such nonlinearities directly into the prediction model. It also bears noting that the tracking is without an offset.

Minimum Time

According to minimum-time scheme, the initial target set was calculated for $T = 2$, which means that the reference will be reached in 2 time steps once the state is contained in the set. Starting from this set the minimum-time controller was calculated as a look-up table consisting of 2993 regions. The performance of this controller is compared with the CFTOC approach in Figure 5. As can be seen from the picture, the minimum-time controller reacts faster and drives the position to the respective reference in less number of steps compared to the CFTOC scheme. The case in Figure 5 represents a switch from the fully-opened to the fully-closed position. For this scenario the minimum-time policy achieves the reference in 10 steps (i.e. 7 seconds), while the CFTOC requires 14 sampling instances (i.e. 7 seconds) for the position to reach the reference with zero angular velocity.

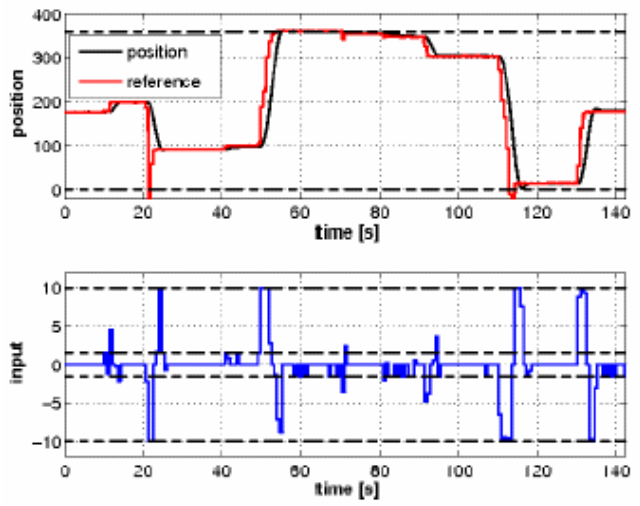

Fig. 4 Position tracking using the CFTOC approach.

Interesting to notice is the time profile of the manipulated variable produced by the minimum-time controller. Although one would expect that the input will reach the constraints immediately after the change of the reference, it is not so. The reason lies in the discrete-time formulation of the problem. Specifically, at the first instance following the change of the reference the controller tries to reach a set of states which is “closer” to the reference. However, due to the discrete-time formulation there may be multiple choices of the control input which drive the system states to a prescribed target set T_{set} in (21). From this multiple choices only the one which minimizes the given performance criterion (21) is chosen. Therefore the control profile doesn’t necessarily have to reach the constraints on the first instance in order to guarantee convergence of the states towards the reference in least possible number of discrete steps.

Conclusion

In this paper a hybrid model of a servo engine is derived, based on the PWA modelling approach. The hybrid model was adopted to cope with a deadzone nonlinearity. Consequently, the validation with experimental data has shown that this approach matches the behavior of the real plant well. Subsequently the model has been used to design an MPC control policy. Two approaches have been considered, a CFTOC scheme in which an optimal control problem of finite size is solved, and a minimum-time setup. The main contribution of the paper is the extension of the minimum-time algorithm to cope with reference tracking problems. The results of the control design is a lookup table which allows one to implement MPCbased controllers in real-time. Both approaches have then been implemented on a target platform, allowing to perform real experiments with low effort.

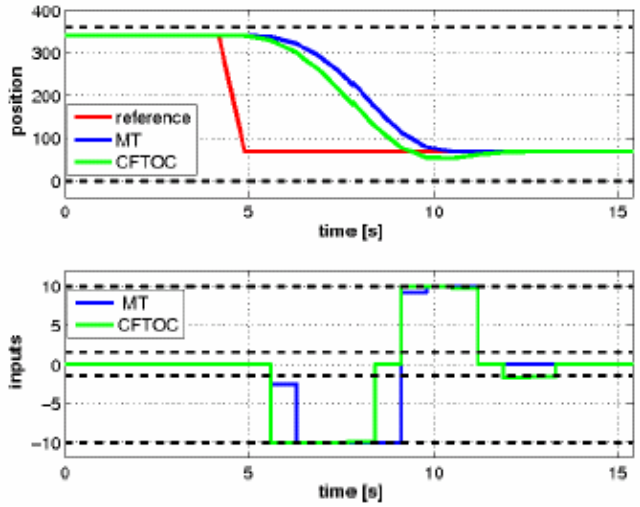

Fig. 5 Comparison between CFTOC and minimum-time control approaches.

Acknowledgment

The authors are pleased to acknowledge the financial support of the Scientific Grant Agency of the Slovak Republic under grants No. 1/3081/06 and 1/4055/07 within the framework of the European Social Fund (PhD Students for Modern Industrial Automation in SR, JPD 3 2005/NP1-047, No 13120200115).

References

[1] ALLGÖWER, F., ZHENG, A., editors.: *Nonlinear Model Predictive Control*. Birkhäuser, 2000.

[2] BEMPORAD, A., MORARI, M.: Control of systemsintegrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, March 1999.

[3] BEMPORAD, A.,MORARI, M., DUA, V., PISTIKOPOULOS, E.N.: The Explicit Linear Quadratic Regulator for Constrained Systems. *Automatica*, 38 (1):3–20, January 2002.

[4] BORRELLI, F.: Constrained Optimal Control of Linear and Hybrid Systems. In *Lecture Notes in Control and Information Sciences*, volume 290. Springer, 2003.

[5] CHOW, C.M., CLARKE, D.: Actuator nonlinearities in predictive control. In *Advances in Model Based Predictive Control*, volume II, pages 80–82, University of Oxford, Department of Engineering Science, 1993.

[6] GAREY, M. R., JOHNSON, D.S.: *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990. ISBN 0716710455.

[7] GRIEDER, P., KVASNICA, M., BAOTIC, M., MORARI, M.: Stabilizing low complexity feedback control of constrained piecewise affine systems. *Automatica*, 41, issue 10:1683–694, October 2005.

[8] HERCEG, M., FIKAR, M.: Constrained Predictive Control of Laboratory Servoengine. In *Proc. of 15th Int. Conference on Process Control 2005*, Štrbské Pleso, Slovakia, June 7-10 2005.

[9] KOUVARITAKIS, B., CANNON, M., editors: *Nonlinear predictive control: theory and practice*. IEE Control Engineering series, 2001.

[10] KVASNICA, M., GRIEDER, P., BAOTIC, M.: Multi-Parametric Toolbox (MPT), 2004. URL http://control.ee.ethz.ch/~mpt/.

[11] MACIEJOWSKI J.M.: *Predictive Control with Constraints*. Prentice Hall, 2002.

[12] PASTOREKOVÁ, L., MÉSZÁROS, A., BURIAN, P.: Intelligent neural controller design in MATLAB environment. In *Proc. 7. International Scientific - Technical Conference PROCESS CONTROL 2006*, page 155, Kouty nad Desnou, Czech Republic, June 13-16, 2006 2006.

[13] TØNDEL, P., JOHANSEN, T.A., BEMPORAD, A.: Evaluation of piecewise affine control via binary search tree. *Automatica*, 39:945–950, 2003.

[14] ČIRKA, Ľ., MIKLEŠ, J., FIKAR, M.: A deterministic LQ tracking problem: Parameterization of the controller. *Kybernetika*, 38(4):469–478, 2002.

[15] ČIRKA, Ľ., FIKAR, M., MIKLEŠ, J.: Adaptive LQ Control of a Laboratory Fan Heater. In *Proc. 7. International Scientific-Technical Conf. Process Control 2006*, page 11, Kouty nad Desnou, Czech Republic, June 13-16, 2006 2006.

Ing. Martin Herceg

Škola: Slovenská Technická Univerzita v Bratislave
Fakulta/Ústav: Fakulta Chemickej a Potravinárskej Technológie, Ústav Automatizácie, Informatizácie a Matematiky
Katedra: Oddelenie Informatizácie a Riadenia Procesov
Ulica: Radlinského 9
PSČ a Mesto: 812 37 Bratislava
Tel.: +421 2 52495269
Fax: +421 2 52496469
E-mail: martin.herceg@stuba.sk