Microcontroller implementation of mixed-integer predictive control

J. Novak and P. Chalupa

***Abstract*—** The development of efficient methods and also technological advances permit the implementation of predictive control on embedded systems with limited computational power and memory capacity. However, predictive control with a finite number of admissible input values still remains restricted to powerful computational platforms as problem becomes NP-hard. Given the growing computational power of embedded controllers the topic becomes more and more attractive. In the paper the solver based on a standard branch-and-bound method and interior point method is implemented the target system with low power and limited RAM memory. The performance is evaluated in two simple simulation experiments. In the first simulation experiments the goal is to control the level in a MIMO water tank with binary, integer and continuous input signals. The second experiment evaluates the predictive control based on the hybrid model of the two-tank system.

***Keywords*—** Embedded Systems, Predictive Control, Mixed-Integer Quadratic Programming, Branch-and-bound Method..

I. Introduction

Model predictive control has had recorded an exceptional growth growth due to its independent adoption by the process industries where it proved to be highly successful in comparison with alternative methods of multivariable control [1]. Its phenomenal success is contributed to its ability to handle hard constraints on inputs and states that arise in most applications and its conceptual simplicity to handle complex systems. Process controls with a finite number of admissible values are common in a large number of relevant applications. For example, chemical plants are equipped with valves that can be either open or closed. A large potential for optimization is found as the number of potential modes of operation is hard to explore in an exhaustive way. The Mixed-Integer Quadratic Control or Hybrid optimal control as referred by several authors [3] addresses the optimal control problem of such systems.

The resulting mixed-integer quadratic problems (MIQP) are usually solved using the commercial solvers such as CPLEX or GUROBI that enables to deploy mixed-integer MPC on desktop platform. The necessity of these solvers limits the application to powerful computing platforms and slow dynamical systems.

Recently, efficient online solution methods for convex quadratic problem have been developed that can be implemented on the embedded hardware and are able to solve the problem at high sampling rate. An online MPC strategy with a good balance between computational speed and memory demand based that uses a fast gradient method was developed in [4]. An interior point solver was specifically designed for embedded applications in [5]. Many real-life problems can be represented as MLD (Mixed Logic Dynamical) systems [3] which are hybrid systems and whose MPC control requires solution of the MIQP problem [6]. Different methods for hybrid optimal control problem solution were evaluated in [7]. Currie, Prince-Pike and Wilson developed a MATLAB framework for generating fast model predictive controllers for embedded targets such as ARM processors and tested it on inverted pendulum in [8]. Bleris and Kothare present a real-time implementation of the MPC on a microcontroller for Glucose regulation in [9]. Implementation aspects of the MPC on Embedded System are also discussed in [10], [11] and [12]. The increase in computational power such as ARM Cortex processors and advances in optimization algorithms has opened a new trend which brings MPC capabilities also to complex and fast systems. With the development of cheap multi-core CPU in microcontrollers, the parallel computation might be the promising way for further decrease of computation time.

The aim of the paper is to illustrate the practical feasibility of mixed-integer MPC with constraints on a low cost embedded system where the problem is solved using branch-and-bound method and the relaxed quadratic programming problem is solved with interior point method. The paper is structured as follows: Section II briefly repeats the MIQP formulation and the branch-and-bound algorithm. The interior point method used for solution of the relaxed problem is described in Section III. The description of the embedded system is given in Section IV. Section V contains the results of the implementation of the solver on embedded system for simulation example of a two-tank system. In Section VI the hybrid model of two tank system is developed using MLD strategy with considering all constraints of the physical plant

The research was supported with financial support of research project NPU I No. LO1303 by the Ministry of Education of the Czech Republic and also by the European Regional Development Fund under the Project CEBIA-Tech No. CZ.1.05/2.1.00/03.0089..

Jakub Novak is with Faculty of Applied Informatics, Tomas Bata University in Zlin, nam. T.G. Masaryka 5555, 76001 Zlin, Czech Republic (email: jnovak@fai.utb.cz)

Petr Chalupa is with Faculty of Applied Informatics Tomas Bata University in Zlin, nam. T.G. Masaryka 5555, 76001 Zlin, Czech Republic (email: chalupa@fai.utb.cz).

and the model is used in mixed-integer predictive control.

. Finally, the main conclusions are summarized in the last section.

II. BRANCH-AND-BOUND METHOD

In this section the description of the implemented branch-and-bound algorithm is given. Mixed-integer quadratic programming (MIQP) problems are optimization problems with a quadratic objective function, subject to linear equality and inequality constraints as presented below and where some variables are constrained to be integers. A common special case of MIQP is when the integer variables are constrained to binary values 0 or 1. The problem is non convex due to the fact that the optimized variables x_i belong to the binary set. The general formulation for a mixed-integer quadratic programming problem is the following,

$$\begin{aligned} &\min 0.5\left[\boldsymbol{x}_c^T, \boldsymbol{x}_i^T\right]\boldsymbol{H}\begin{bmatrix}\boldsymbol{x}_c^T\\ \boldsymbol{x}_i^T\end{bmatrix} + f^T\begin{bmatrix}\boldsymbol{x}_c^T\\ \boldsymbol{x}_i^T\end{bmatrix} \\ &x_c \in R^{n_c}, x_i \in N^{n_i} : \boldsymbol{a}_j^T\begin{bmatrix}\boldsymbol{x}_c^T\\ \boldsymbol{x}_i^T\end{bmatrix} + b_j = 0, \quad j = 1,\dots,m_{ec} \\ &\qquad \boldsymbol{c}_j^T\begin{bmatrix}\boldsymbol{x}_c^T\\ \boldsymbol{x}_i^T\end{bmatrix} + d_j \geq 0, \quad j = 1,\dots,m_{ic} \end{aligned} \tag{1}$$

where $\boldsymbol{H}$ is a positive definite n x n matrix ($n = n_i + n_c$), f is the n-dimensional vector. The n-dimensional vectors $\mathbf{a_j}$ and $\mathbf{c_j}$ and vectors $\boldsymbol{b}$ and $\boldsymbol{d}$ are used to set up the constraints. The numbers of equality and inequality constraints are specified with m_{ec} and m_{ic}, respectively. The equality and inequality constraints define a feasible region in which the solution to the problem must be located in order for the constraints to be satisfied. The only difference when compared to the convex QP is the presence of binary variables $\boldsymbol{x_i}$. Fortunately, if the binary variable is fixed or relaxed, a convex set is obtained and the problem can be solved using conventional methods for convex optimization. A constrained QP is usually solved either using an interior point method or an active set method.

Branch-and-bound has been the most used tool for solving large scale *NP*-hard combinatorial optimization problems since the branch-and-bound method is an order of magnitude faster than any of the other methods such as Generalized Benders Decomposition or Outer Approximation. The method is so fast due to the fact that that the QP subproblems are easy to solve. For MATLAB, free software like YALMIP [13] can be used. During the solution process, the status of the solution is described by a pool of yet unexplored subset of the solution space and the best solution found so far. The nodes in a dynamically generated search tree, which initially only contains the root, and each iteration of a classical branch-and-bound algorithm processes one such node represent unexplored subspaces. The iteration has two main components: selection of the node to process and branching strategy. The nodes created are then stored together with the bound of the currently processed node. The search stops when the pool of unexplored subset is empty and the optimal solution is then the one recorded as "current best".

There are two common node selection strategies for selection of the node to proceed in the next iteration. The first one is best-first-search, where the next node is always the one with the lowest dual bound. This method however requires a large amount of storage. The second class of node selection strategies depth-first-search where warm-starting can be successfully applied due to the similarity of the subproblems and also number of unexplored nodes is low, which significantly reduces the storage requirements. Due to the limited memory of the microcontroller the depth-first-search strategy is used in the example. Branching on a variable involves choosing the branching variable of the current optimal solution of the relaxed problem and then adding a constraint to it. The maximum fractional branching strategy which chooses the variable with the highest fractional part is used in the solver. The scheme of branch-and-bound method is depicted in Fig. 1.

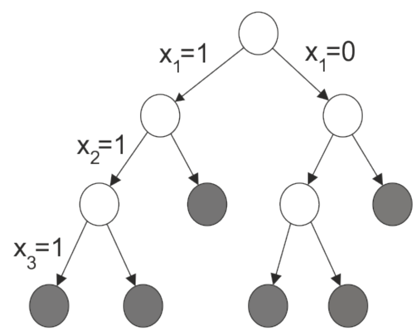

Fig. 1 branch-and-bound method

III. INTERIOR POINT METHOD

At each node the relaxed QP is solved solved either using an interior point method or an active set method. Interior-point methods solve problems iteratively where each iteration is computationally expensive but can make significant progress towards the solution. The solver uses the interior point method for solution of the relaxed problems:

$$\begin{aligned} &\min 0.5\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} + \boldsymbol{f}^T\boldsymbol{x} \\ &x \in R^n : \boldsymbol{A}_c\boldsymbol{x} + \boldsymbol{b}_c = 0 \\ &\qquad \boldsymbol{C}_c\boldsymbol{x} + \boldsymbol{d}_c \geq 0 \end{aligned} \tag{2}$$

where $\boldsymbol{A}$ is a m_{ec} *x n* matrix describing the equality constraints and C is an mic x n matrix describing the inequality constraints. $\boldsymbol{b}$ and $\boldsymbol{d}$ are m_{ec} *x n* and m_{ic} *x n* vectors respectively. The Lagrangian $\boldsymbol{L}$(x,y,z) with vectors $\boldsymbol{y}$ and $\boldsymbol{z}$ containing the Lagrange multipliers is defined as:

$$L(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} + \boldsymbol{f}^T\boldsymbol{x} - \boldsymbol{y}^T(\boldsymbol{A}_c\boldsymbol{x} - \boldsymbol{b}_c) - \boldsymbol{z}^T(\boldsymbol{C}_c{}^T - \boldsymbol{d}_c) \qquad (3)$$

The following optimality conditions can be obtained with the introduction of the slack vector :

$$\begin{aligned} &\boldsymbol{H}x + \boldsymbol{f} - \boldsymbol{A}_c{}^T - \boldsymbol{C}_c\boldsymbol{z} = 0 \\ &\boldsymbol{A}_c\boldsymbol{x} - \boldsymbol{b}_c = 0 \\ &\boldsymbol{s} - \boldsymbol{C}_c\boldsymbol{x} + \boldsymbol{d}_c = 0 \\ &(\boldsymbol{z},\boldsymbol{s}) \geq 0 \\ &s_i\, z_i = 0 \end{aligned} \qquad (4)$$

Defining the function $\boldsymbol{F}($x,y,z,s) such that the roots of this function are solutions to the first four optimality conditions we obtain set of linear equation:

$$\begin{bmatrix} \boldsymbol{G} & -\boldsymbol{A}_c{}^T & -\boldsymbol{C}_c{}^T & 0 \\ -\boldsymbol{A}_c & 0 & 0 & 0 \\ -\boldsymbol{C}_c & 0 & 0 & \boldsymbol{I} \\ 0 & 0 & \boldsymbol{S} & \boldsymbol{Z} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ \Delta s \end{bmatrix} = - \begin{bmatrix} \boldsymbol{H}x + \boldsymbol{f} - \boldsymbol{A}_c{}^T\boldsymbol{y} - \boldsymbol{C}_c{}^T\boldsymbol{z} \\ \boldsymbol{A}_c\boldsymbol{x} - \boldsymbol{b}_c \\ \boldsymbol{s} - \boldsymbol{C}_c\boldsymbol{x} + \boldsymbol{d}_c \\ (\mathrm{s}^*\mathrm{z}) \end{bmatrix} \qquad (5)$$

where * is the element-wise multiplication of vectors. For solution of this set of equation predictor-corrector method proposed by Mehrotra is used. As a stopping criterion the following criterions are used:

$$\begin{aligned} &\left\|\boldsymbol{H}x + \boldsymbol{f} - \boldsymbol{A}_c{}^T\boldsymbol{y} - \boldsymbol{C}_c{}^T\boldsymbol{z}\right\| \leq \varepsilon \\ &\left\|\boldsymbol{A}_c\boldsymbol{x} - \boldsymbol{b}_c\right\| \leq \varepsilon \\ &\left\|\boldsymbol{s} - \boldsymbol{C}_c\boldsymbol{x} + \boldsymbol{d}_c\right\| \leq \varepsilon \\ &\left\|\mu\right\| \leq \varepsilon \end{aligned} \qquad (6)$$

and also maximum number of iterations k_{max} is specified. For the solution of the set of linear equations $\boldsymbol{Ax=b}$ from (5) the LDLT factorization is used.

$$\boldsymbol{PAP}^T = \boldsymbol{LDL}^T \qquad (7)$$

where $\boldsymbol{P}$ is a permutation matrix, $\boldsymbol{L}$ is a unit lower triangular matrix and $\boldsymbol{D}$ is a block diagonal matrix with 1x1 and 2x2 blocks. Once a factorization has been computed, the solution to the linear system $\boldsymbol{Ax} = \boldsymbol{b}$ can be computed at comparably low cost by solving a sequence of equations:

$$\begin{aligned} &\boldsymbol{Lu} = \boldsymbol{Pb} \\ &\boldsymbol{Dv} = \boldsymbol{u} \\ &\boldsymbol{L}^T\boldsymbol{w} = \boldsymbol{v} \\ &\boldsymbol{x} = \boldsymbol{P}^T\boldsymbol{w} \end{aligned} \qquad (8)$$

with intermediate vectors ***u,v,w***. The cost of the solve procedure (8) is most of the time negligible with respect to the cost of computing the factorization (7). The LDL factorization implemented in the LAPACK library [14] exploits the partial pivoting based on the Bunch-Kaufmann method [15].

IV. SELECTED HARDWARE PLATFORM

The proposed MIQP problem solver was implemented on The Stellaris® LM4F120 board which is a low-cost evaluation platform for 32-bit ARM® Cortex™-M4F-based microcontrollers from Texas Instruments (Fig 3).

The microcontroller runs at 80 MHz. The board has 32KB of SRAM memory, 256KB of flash memory and 2KB EEPROM. For implementation of the solver the requirements for memory and evaluation speed must be considered. The board has only 32KB of RAM however system parameters and constraints can be stored in flash memory as they are fixed and only read during the solution of the problem.

We have developed a simple implementation of branch-and-bound algorithm with interior point method for computation of the relaxed problem, written in C, using the LAPACK and BLAS libraries to carry out the numerical linear algebra computations such as matrix-vector multiplication, LDLT decomposition and solution of system of algebraic equations. The solver is implemented using double precision floating-point arithmetic.

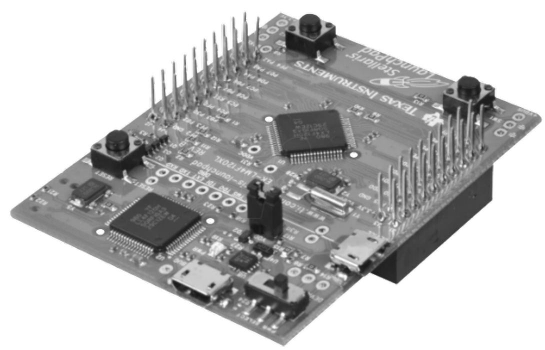

Fig. 3 Stellaris LM4F120 launchpad board

V. NUMERICAL EXAMPLE I

The first numerical example consists of the buffer and supply tanks. There are four control inputs: a two-stage pump, a continuous heater, and two on/off valves . The function of the plant is to receive liquid from an upstream process, and to deliver this liquid at some reference temperature to a downstream process.

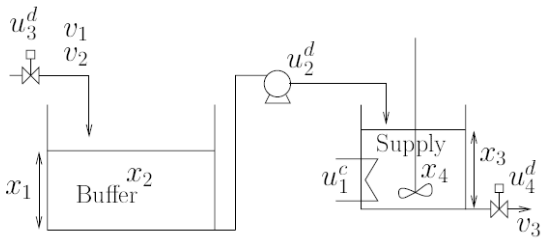

Fig. 2 two-tank system

The process dynamics is given by the following set of differential equations:

$$\begin{aligned}
\dot{x}_1 &= \frac{1}{A_b}\left(v_1 u_3^d - \alpha u_2^d\right) \\
\dot{x}_2 &= \frac{1}{A_b x_1}\left(-x_2 v_1 u_3^d + v_1 v_2 u_3^d\right) \\
\dot{x}_3 &= \frac{1}{A_s}\left(\alpha u_2^d - v_3 u_4^d\right) \\
\dot{x}_4 &= \frac{1}{A_s x_3}\left(\left(x_2 - x_4\right)\alpha u_2^d + \frac{u_1^c}{c_l \rho_l}\right)
\end{aligned} \tag{9}$$

where $x = \left[x_1\ x_2\ x_3\ x_4\right]^T$ is the state, $u = \left[u_1^d\ u_2^d\ u_3^d\ u_4^c\right]^T$ is the control input, and $v = \left[v_1\ v_2\ v_3\ \right]^T$ is the disturbance signal. The model parameters are given in Table I, the legend for the states, controls, and disturbances is given in Table II. The system was linearized at steady-state point $x^s = [7m, 18\,^0C, 1.5m, 22\,^0C]^T$ for input signal $u^s = [280W, 1, 1, 1]^T$. Maximal output of the heater is 560W.

Table I Model parameters

A_b	3.5 m^2	Buffer area
A_s	2 m^2	Supply area
c_l	4.2kJ/kgK	Specific liquid heat capacity
ρ_l	1000kg/m^3	Liquid density
α	1m^3/min	Pump capacity factor
Δ	0.25min	Sample period

Table II States, controls and disturbances

x_1,x_2	Buffer and supply levels
x_2,x_4	Buffer and supply temperatures
u_1^c	Heater
u_2^d	Pump
u_3^d, u_4^d	Inlet and outlet valves
v_1	Inflow
v_2	Temperature of inflow
v_3	Outflow

The mixed integer predictive control optimization problem is based on a time-invariant discrete process model and linear constraints:

$$\begin{aligned}
&\min_{\Delta u(\mathrm{k}), \Delta u(\mathrm{k+1}), \ldots, \Delta u(\mathrm{k+N_c-1}),} J(\mathrm{k}) \\
&J(\mathrm{k}) = \sum_{i=1}^{N_p}\left(\hat{y}(\mathrm{k+i}) - y_r(\mathrm{k+1})\right)^T Q\left(y(\mathrm{k+i}) - y_r(\mathrm{k+1})\right) + \\
&\qquad \sum_{i=1}^{N_c} \Delta u\left(\mathrm{k+i-1}\right)^T R \Delta u\left(\mathrm{k+i-1}\right) \\
&\textit{subject to}:\quad A_c \Delta u \le b_c
\end{aligned} \tag{10}$$

where $\hat{y}(\mathrm{k}+i)$ is the ith step output prediction, $y_r(\mathrm{k+i})$ is the *i*-th step of the reference trajectory, $\boldsymbol{\Delta u(k)}$ is difference between $\boldsymbol{u(k)}$ and $\boldsymbol{u(k\text{-}1)}$, $\boldsymbol{R,Q}$ are positive definite matrices and N_p and N_c are the prediction and control horizons, respectively and $\boldsymbol{A_c}$ and $\boldsymbol{b_c}$ are the constraints matrix and vector that can be derived from process model and input and state constraints. Only the first element of the optimal predicted input sequence $\Delta\boldsymbol{u}(k)$ is applied to the plant:

$$u(\mathrm{k}) = \mathrm{u(k-1)} + \Delta u(\mathrm{k}) \tag{11}$$

The normalized discrete linearized model of the plant is assumed in the form:

$$\begin{aligned}
\boldsymbol{x}(\mathrm{k+1}) &= \boldsymbol{A}\boldsymbol{x}(\mathrm{k}) + \boldsymbol{B}\boldsymbol{u}(\mathrm{k}) \\
\boldsymbol{y}(\mathrm{k}) &= \boldsymbol{C}\boldsymbol{x}(\mathrm{k})
\end{aligned} \tag{12}$$

where $\boldsymbol{u}(k)$ is the vector of manipulated variables or input variables; $\boldsymbol{y}(k)$ is the vector of the process outputs and $\boldsymbol{x}(k)$ is the state variable vector. The sampling time was set to 15s. Using the linear model the model predictive controller would exhibit steady – state offset in the presence of plant/model mismatch or unmeasured disturbance due to lack of integral action. In order to introduce integral behavior, a new state variable vector is chosen to be:

$$\boldsymbol{x} = \begin{bmatrix} \Delta\boldsymbol{x}(\mathrm{k}) \\ \boldsymbol{y}(\mathrm{k}) \end{bmatrix} \tag{13}$$

Combining (12) and (13) leads to the following state-space model:

$$\begin{aligned}
\begin{bmatrix} \Delta\,\mathrm{x(k+1)} \\ y(\mathrm{k+1}) \end{bmatrix} &= \begin{bmatrix} \boldsymbol{A} & 0 \\ \boldsymbol{CA} & I \end{bmatrix}\begin{bmatrix} \Delta\,\mathrm{x(k)} \\ y(\mathrm{k}) \end{bmatrix} + \begin{bmatrix} \boldsymbol{B} \\ \boldsymbol{CB} \end{bmatrix}\Delta u(\mathrm{k}) \\
y(\mathrm{k}) &= \begin{bmatrix} 0 & 1 \end{bmatrix}\begin{bmatrix} \Delta\,\mathrm{x(k)} \\ y(\mathrm{k}) \end{bmatrix}
\end{aligned} \tag{14}$$

The process state-space model can be rewritten as a prediction model for the current state vector $\boldsymbol{x}$ and control increment sequence $\Delta\boldsymbol{U} = \left[\Delta u(\mathrm{k}), \Delta u(\mathrm{k+1}), \ldots, \Delta u(\mathrm{k}+N_c-1)\right]^T$. For given horizons it can be formulated in terms of vectors as:

$$\boldsymbol{Y} = \boldsymbol{K}\boldsymbol{x} + \boldsymbol{L}\Delta\boldsymbol{U} \tag{15}$$

where $\boldsymbol{Y} = \left[y(\mathrm{k}+1), \mathrm{y}(\mathrm{k}+2), ..., \mathrm{y} = \mathrm{k}+\mathrm{N_p} \right]^T$ and $\boldsymbol{K}$ and $\boldsymbol{L}$ are constant matrices derived from the process model.

The Hessian matrix $\boldsymbol{H}$ and vector $\boldsymbol{f}$ from criterion (1) can then be formulated as:

$$\boldsymbol{H} = \boldsymbol{L}^T \boldsymbol{Q} \boldsymbol{L} + \boldsymbol{R},\ f = -\boldsymbol{R}(\boldsymbol{Y}_\mathrm{r} - \boldsymbol{K}\boldsymbol{x}(\mathrm{k}))^T \boldsymbol{L} \tag{16}$$

The control task is to keep the supply temperature at its nominal value while preventing overflow/emptying of the buffer and supply tank. Both the prediction and control horizons were set to 4 steps. The mathematical formulation of the predictive control for prediction and control horizon of 4 steps with constraints results corresponds to an MIQP with 4 continuous variables (Δu_1) and 8 binary variables ($\Delta u_3, \Delta u_4$) and 4 integer variables (Δu_2) and 48 inequality constraints. The weighting matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ were set identity matrices. Sixteen of the 48 inequality constraints are necessary to restrict the 8 binary values from 0 to 1. The execution times and number of relaxed QP solved at each sampling point is presented in Fig. 4. The closed-loop response of the MPC controller is presented in Fig. 5 and 6.

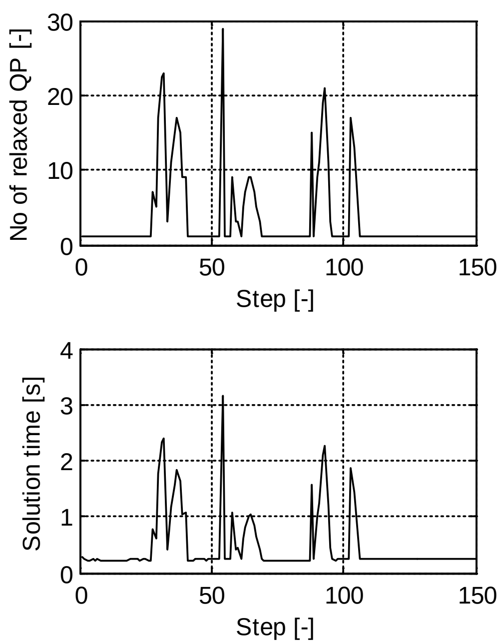

Fig. 5 number of solved relaxed quadratic problems and execution time

The matrices $H \in R^{nxn}$, $C_c \in R^{n_{ic}xn}$, $A_c \in R^n$ and vectors *f*, b_c, d_c for definition of constraints and cost function of the MIQP problem are stored in flash memory. The branch-and-bound method requires a pool for storing the nodes during the solution process. The memory requirements in bytes are given by two matrices for storage the additional constraints of the size n_{pool} * n_i , a double vector of the size n_{pool} to hold the bounds for each node in a pool and integer vector to store the priority of the nodes in the pool. Interior point method requires allocation of vectors $x, \Delta x, y, \Delta y, \mathrm{z}, \Delta z, s, \Delta s$, the matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$ of the system $\boldsymbol{Ax=b}$ in the memory. The matrix A is symmetric so only lower triangular part of the matrix A is stored. The number of elements of lower triangular matrix is given as:

$$\frac{1}{2}(\mathrm{n}+\mathrm{n}_{\mathrm{ic}}+\mathrm{n}_{ec})(\mathrm{n}+\mathrm{n}_{\mathrm{i}c}+\mathrm{n}_{ec}+1) \tag{17}$$

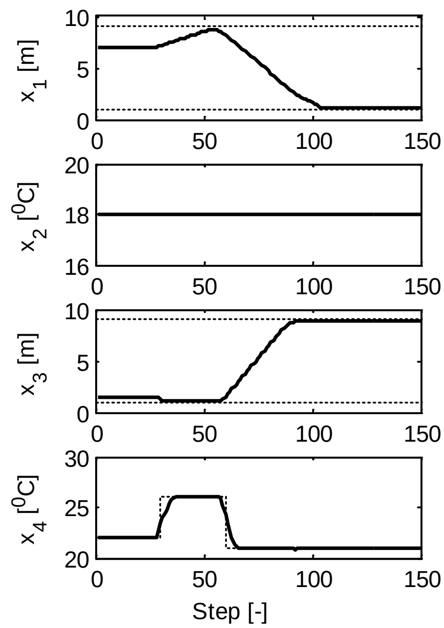

Fig. 5 closed-loop response of the two-tank system using mixed-integer predictive controller – system states (dotted line –system constraint and reference signal for x_4)

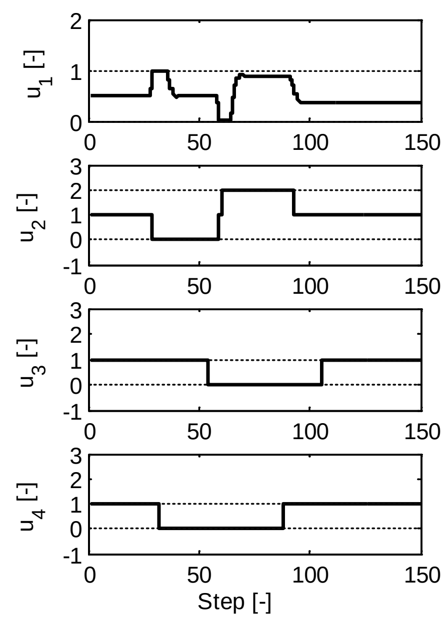

Fig. 6 closed-loop response of the two-tank system using mixed-integer predictive controller – system inputs (dotted line- input constraints)

VI. NUMERICAL EXAMPLE II

In the second example the same optimization method is used for control of two-tank system using the MLD modelling framework. The MLD modeling framework is based on the idea of translating logic relations, discrete/logic dynamics, A/D (analog to digital (logic)), D/A conversion and logic constraints into mixed integer linear inequalities. These inequalities are combined with the continuous dynamical part, which is described by linear difference equations. The resulting MLD system is described by the following relations:

$$\begin{aligned} &x(\mathrm{k}+1) = \mathrm{A}\, x(\mathrm{k}) + \mathrm{B}_1\, u(\mathrm{k}) + \mathrm{B}_2\, \delta(\mathrm{k}) + \mathrm{B}_3\, z(\mathrm{k}) + \mathrm{B}_0 \\ &\mathrm{y}(\mathrm{k}) = Cx(\mathrm{k}) + D_1 u(\mathrm{k}) + D_2\delta(\mathrm{k}) + D_3 z(\mathrm{k}) + D_0 \\ &E_2\delta(\mathrm{k}) + \mathrm{E}_3\, z(\mathrm{k}) \le \mathrm{A}\, x(\mathrm{k}) + E_1 u(\mathrm{k}) + E_4 x(\mathrm{k}) + E_5 \end{aligned} \tag{18}$$

where x is the vector of states which can be continuous or binary, u is the command input and may also contain continuous and binary commands, δ and z are respectively auxiliary logical and continuous variables, and y is the system output.

In [3], Bemporad and Morari introduced a model predictive control of hybrid systems using mixed logical dynamical (MLD) system description and a mixed integer linear program solver. Assuming a quadratic cost function from the MLD model the optimization has the following form:

$$\min J(u,\mathrm{z},\delta) = \sum_{k=0}^{N-1} \begin{aligned} &\|u(\mathrm{k}) - \mathrm{u}_e\|^2_{Q_1} + \|\mathrm{z}(\mathrm{k}) - z_e\|^2_{Q_2} + \\ &+ \|\delta(\mathrm{k}) - \delta_e\|^2_{Q_3} + \|y(\mathrm{k}) - w(k)\|^2_{Q_4} \end{aligned} \tag{19}$$

Subjected to equations which define the MLD system for N steps ahead predictions (1). The matrices $Q_{1,} Q_{2,} Q_{3,} Q_{4,}$ are given weight matrices. This problem can be rewritten to a standard MIQP programing form:

$$\begin{aligned} &\min 0.5\left[\boldsymbol{u}_c^T, \boldsymbol{u}_i^T\right] \boldsymbol{H} \begin{bmatrix} \boldsymbol{u}_c^T \\ \boldsymbol{u}_i^T \end{bmatrix} + f^T \begin{bmatrix} \boldsymbol{u}_c^T \\ \boldsymbol{u}_i^T \end{bmatrix} \\ &u_c \in R^{n_c}, u_i \in N^{n_i} : \boldsymbol{a}_j^T \begin{bmatrix} \boldsymbol{u}_c^T \\ \boldsymbol{u}_i^T \end{bmatrix} + b_j = 0, \quad j = 1,\dots,m_{ec} \\ &\qquad \boldsymbol{c}_j^T \begin{bmatrix} \boldsymbol{u}_c^T \\ \boldsymbol{u}_i^T \end{bmatrix} + d_j \ge 0, \quad j = 1,\dots,m_{ic} \end{aligned} \tag{20}$$

where n_c and n_i define the numbers of continuous and integer variables, $\boldsymbol{H}$ is a positive definite matrix, $\boldsymbol{f}$ is the n-dimensional vector. The n-dimensional vectors a_j and c_j and vectors b and d are used to set up the constraints. The numbers of equality and inequality constraints are specified with m_{ec} and m_{ic}, respectively. The equality and inequality constraints define a feasible region in which the solution to the problem must be located in order for the constraints to be satisfied. The optimization procedure of (20) leads to problems with the following optimization vector :

$$\boldsymbol{u} = \begin{bmatrix} u(\mathrm{k}),\dots,\mathrm{u}(\mathrm{k}+\mathrm{N}-1), \delta(\mathrm{k}),\dots,\delta(\mathrm{k}+\mathrm{N}-1), \\ \mathrm{z}(\mathrm{k}),\dots,\mathrm{z}(\mathrm{k}+\mathrm{N}-1) \end{bmatrix} \tag{21}$$

The predictor matrices for future outputs of the system were recursively developed using formulas in .

$$\hat{\boldsymbol{Y}} = \boldsymbol{K} * x(k) + \boldsymbol{Lu} \tag{22}$$

The parameters of the simulation model were taken from [17] where hybrid modeling of such system is considered. The system consists of two liquid tanks that can be filled with pump acting on the tank 1 (Fig. 7.). The pump delivers the liquid with flow-rate Q_l with upper limit Q_{rnax} and represents continuous input u_1. The tanks are interconnected to each other through two pipes. The liquid levels h_1 and h_2 in each tank can be measured with continuous valued level. The valve V_1 is kept constantly open. The valve V_2 can be opened or closed and it is considered as discrete input u_2. The control task is to control level in the second tank.

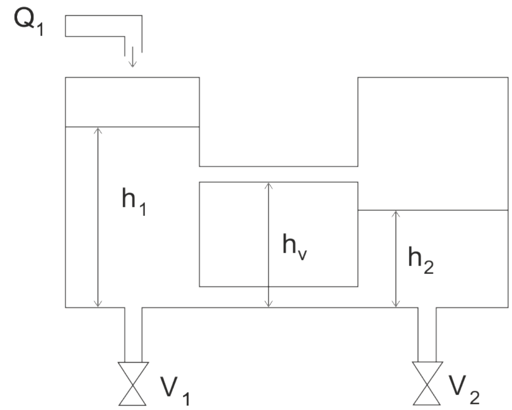

Fig. 7 Two-tank system scheme)

The MLD model was developed using the HYSDEL Toolbox [18]. The resulting model has two states $x = [h_1, h_2]$, the input vector combines continuous and discrete part $u = [Q_1, V_2]$ and there are two continuous auxiliary variables $z = [z_1, z_2]$ which define the flow-rate through the upper pipe and flow-rate through the valve V_2. The auxiliary binary variable δ reflects whether level in the first tank is higher than h_v.

Table III Model parameters

S	00154 m^2	Tank Section
A_s	3.6E-5 m^2	Cross-section area
c_l	9.81m/s^2	Gravity constant
ρ_l	0.5m	Height of pipe
Q_{max}	2E-4m^3/s	Maximum inflow
Δ	5s	Sampling time

The prediction horizon equal to 4 steps is used with a sampling time Ts = 5 s. Constraints on the first control input are added:

$$0 \le u_1 \le Q_{\max} \tag{23}$$

The weighting factors in (19) were set to :

$$Q_1 = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.0001 \end{bmatrix}$$
$$Q_2 = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.01 \end{bmatrix} \tag{24}$$
$$Q_3 = 0$$
$$Q_4 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Increasing the horizons may increase the performance of the controller, but it increases the complexity of the mixed-integer programming procedure. The choice of prediction horizon offers reasonable values in terms of performance and speed.

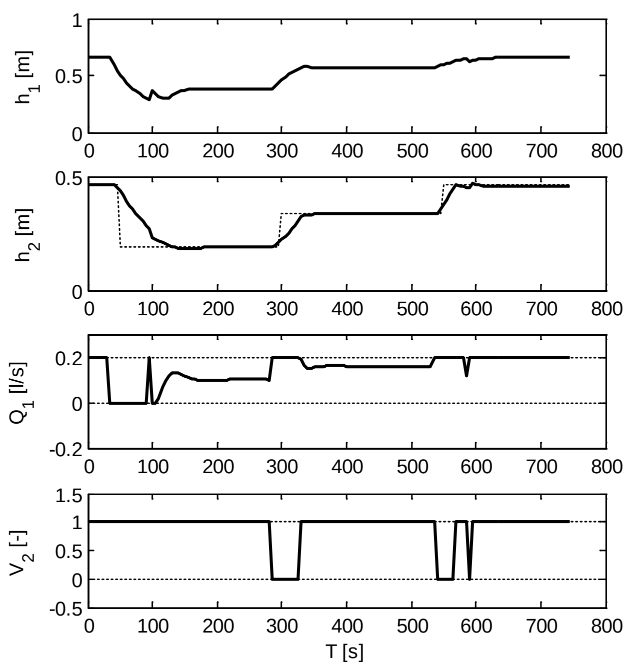

Fig. 8 Mixed-integer predictive control with hybrid model

The increased complexity given by the MLD description of the system with prediction horizon of 4 steps resulted in the optimization problem with 20 optimization variables and 76 inequalities.

$$\begin{aligned} m_{ic} &= 19 * H_p \\ n_c &= 3 * Hp \\ n_i &= 2 * Hp \end{aligned} \tag{25}$$

The closed-loop response of the MPC controller is presented in Fig. 8.

VII. Conclusion

The results show that solution of the predictive control problem with 16 variables (4 continuous, 4 integer and 8 binary) and 48 constraints is manageable on the low power platform. Enough free time remains for the control loop including Kalman filter for state estimation and filtration. Mixed Logical Dynamical (MLD) approach appears as a systematic way for modeling hybrid system, where logic conditions can be easily transformed into mixed-integer inequalities. On the other hand it increases the computational complexity of the optimization problem and limits the application for predictive control with short prediction horizon. The simulation study showed that the memory as well as the computational demand of an MIQP solver implementation is decisive for real-time use on low-cost embedded systems.

References

[1] S.J. Qin and B.J. Badgwell, “A survey of industrial model predictive control technology,” *Control Engineering Practice*, vol. 11, pp. 733–764, 2003.

[2] J.M. Maciejowski, *Predictive Control with Constraints*, Prentice Hall, 2000.

[3] A. Bemporad and M. Morari, “Control of systems integrating logic dynamics and con-straints,” *Automatica* , vol. 35, no. 3, pp. 407–427, 1999.

[4] Y. Wang and S. Boyd, “Fast model predictive control using online optimization,” *IEEE Transactions on Control Systems Technology* vol. 18, pp. 267-278, 2010.

[5] A. Domahidi, E. Chu and S. Boyd, ”ECOS: An SOCP solver for embedded systems,” in *Proc. European Control Conference*, pp. 3071-3076, 2013.

[6] Y. Hammi, N. Zanzouri and M. Ksouri, “Modeling and Predictive Control of Tow Tank System by MLD Approach,“ in *Proc. 5th International Conference on Circuits, Systems, Control, Signals*, 2010, pp. 140-145.

[7] R.T. Meyer, M. Žefran and R. A. DeCarlo, “A comparison of the embedding method with multiparametric programming, mixed-integer programming, gradient-descent, and hybrid minimum principle-based methods, *IEEE Transactions on Control Systems Technology*, vol. 22 pp. 1784-1800, 2014.

[8] J. Currie, A. Prince-Pike and D.I. Wilson, “Auto-code generation for fast embedded model predictive controllers,” in *Proc. International Conference on Mechatronics and Machine Vision in Practice* , pp. 122-128, 2012.

[9] L.G. Bleris and M. V. Kothare, “Implementation of Model Predictive Control for Glucose Regulation on a General Purpose Microprocessor,”

in *Proc. 44th IEEE Conference on Decision and Control*, Seville, pp. 5162-5168, 2005.

[10] P. Zometa, M. Kogel, T. Faulwasser and R. Findeisen, “Implementation aspects of model predictive control for embedded systems,” i*n Proc. the American Control Conference*, pp. 1205-1210, 2012.

[11] B. Huyck, H. J. Ferreau, M. Diehl, J. De Brabanter, J. F. M. Van Impe, B. De Moor and F. Logist, “Towards Online Model Predictive Control on a Programmable Logic Controller: Practical Considerations,” *Mathematical Problems in Engineering* , vol. 2012, 2012.

[12] J. Novak and P. Chalupa, “Implementation Aspects of Embedded MPC with Fast Gradient Method,” *International Journal of Circuits, Systems and Signal Processing*, vol. 8, 504-511, 2014.

[13] J. Lofberg, “YALMIP : a toolbox for modeling and optimization in MATLAB,” in *Proc. IEEE International Symposium on Computer Aided Control Systems Design*, pp. 284-289, 2004.

[14] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney and D. Sorensen, *LAPACK Users' Guide*. Warrendale, PA: SIAM, 1999.

[15] J.R. Bunch and L. Kaufman, “Some stable methods for calculating inertia and solving symmetric linear systems,” *Math. Comp.*, vol. 31, pp. 163–179, 1977.

[16] M. Kubalcik, V. Bobal, “Adaptive Predictive Control of Three-Tank System”, International Journal of Mathematical Models and Methods In Applied Sciences,Vol. 7, No.2, pp. 157-165, 2013

[17] Y. Hammi, N. Zanzouri and M. Ksouri, “Model predictive control of hybrid systems with discrete and continuous inputs, Proceedings of the International Conference on Communications, Computing and Control Applications (CCCA), 2011.

[18] F. Torrisi and A. Bemporad, “HYSDEL – a tool for generating computational hybrid models for analysis and synthesis problems, IEEE Transactions on Control Systems Technology, vol. 12m no. 2, pp. 235-249 ,2004

Jakub Novak was born in Zlín, Czech Republic in 1978. He is a Researcher at Faculty of Applied Informatics of Tomas Bata University in Zlín, Czech Republic. He graduated from Faculty of Technology of the same university with an MSc in Automation and Control Engineering in 2002 and he received a PhD in Technical Cybernetics from Faculty of Applied Informatics in 2007. He is a researcher at the CEBIA-Tech research center at Tomas Bata University in Zlin. His research interests are modeling and predictive control of the nonlinear systems.

Petr Chalupa graduated in 1999 from Brno University of Technology and received the Ph.D. degree in Technical cybernetics from Tomas Bata University in Zlin in 2003. He is a researcher at the CEBIA-Tech research center at Tomas Bata University in Zlin. His professional interests are adaptive and predictive control of real-time systems.