International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:5, No:5, 2011

A Microcontroller Implementation of Model Predictive Control

Amira Abbes Kheriji, Faouzi Bouani, Mekki Ksouri, and Mohamed Ben Ahmed,

Abstract—Model Predictive Control (MPC) is increasingly being proposed for real time applications and embedded systems. However comparing to PID controller, the implementation of the MPC in miniaturized devices like Field Programmable Gate Arrays (FPGA) and microcontrollers has historically been very small scale due to its complexity in implementation and its computation time requirement. At the same time, such embedded technologies have become an enabler for future manufacturing enterprises as well as a transformer of organizations and markets. Recently, advances in microelectronics and software allow such technique to be implemented in embedded systems. In this work, we take advantage of these recent advances in this area in the deployment of one of the most studied and applied control technique in the industrial engineering. In fact in this paper, we propose an efficient framework for implementation of Generalized Predictive Control (GPC) in the performed STM32 microcontroller. The STM32 keil starter kit based on a JTAG interface and the STM32 board was used to implement the proposed GPC firmware. Besides the GPC, the PID anti windup algorithm was also implemented using Keil development tools designed for ARM processor-based microcontroller devices and working with C/C++ langage. A performances comparison study was done between both firmwares. This performances study show good execution speed and low computational burden. These results encourage to develop simple predictive algorithms to be programmed in industrial standard hardware. The main features of the proposed framework are illustrated through two examples and compared with the anti windup PID controller.

Keywords—Embedded systems, Model Predictive Control, microcontroller, Keil tool.

I. Introduction

MOdel Predictive Control has become a mature control strategy in the last few years. The reason of this success can be attributed to the enhanced input and/or state and/or output constraint handling, nonlinear processes handling and the ease extension to the multivariable case. All these advantages make this control strategy attractive to the academic community. In order to predict the future behavior of process output, we must have a model. Mathematical models, especially control models can only describe the dynamics of a physical process in an approximate way [1]. Therefore, the chosen model must be capable of capturing the process dynamics so as to precisely predict the future outputs [2]. [3] and [4] have used the uncertain impulse response robust model predictive control in the robust predictive control. However, this representation presents the drawback of dealing only with open loop stable systems. Another representation is the transfer function especially the CARIMA model which allows controlling the unstable open loop system. The disadvantage of transfer function models is that their use in the multivariable case can be somewhat cumbersome and they are non minimal representations [5]. In this work, the state space model is considered to compute the control signal. The advantage of this type of model is that the multivariable systems can easily be dealt with. Moreover, extensive amount of literature consider this representation to solve robust model predictive control [6], [7], [8], [1], [9], [10], [11], [12].

MPC has been applied in high bandwidth applications such as the slow dynamical systems encountered in chemical process control as well as servomechanisms. In addition, process units as fluid catalic cracking and crude atmospheric distillation have been controlled by MPC controllers for more than two decades [3]. However, in control systems with fast sampling times the MPC has not been introduced yet. But this control strategy is very desirable for applications with fast dynamical systems. Such applications require fast and power devices such as microcontroller able to compute huge matrix operations and to solve on line optimization problem in case of presence of constraints.

A microcontroller is described as a computer on a chip because it contains all the features of a full computer including central processor, volatile and non-volatile memories, input and output ports with special features such as serial communication, analog-to-digital conversion and, more recently, signal processing. The presence of microcontroller in semi conductor products is becoming undoubtedly noticeable. This device is used for a variety of industrial applications such as for medicine and bioengineering, aerospace, automotive systems and transportation, microwave ovens, washing machines, integrated secure network systems, etc. Moreover, the advancement of microcontrollers and what they offer combined with their speed, made them more suitable for a large variety of control applications. There are few recently works which investigated the MPC in fast devices [13], [14], [15]. Until recently, low power consumption and high speed were considered priority requirement for embedded systems.

In this paper, we propose an efficient Model Predictive Controller firmware for a performed STMicroelectronics microcontroller (STM32). The control application could benefit from the power features and flexibility of the STM32F103xB devices. The proposed framework was developed using Keil development tools designed for ARM processor-based microcontroller devices.

The outline of this paper is as follows: in section II, a theoretical background which consists of a review of the GPC method is presented. Section III states the hardware as well

A. Kheriji, F. Bouani and M. Ksouri are with the Department of Electrical Engineering, National School of Engineering of Tunis, Tunisia
M. Ben Ahmed is with STMicroelectronics of Tunis.

as the software development tools used in this application. In section IV, a detailed description of the proposed firmware development is presented in which the different source files and functions are illustrated. In section V, the effectiveness of the proposed code is outlined through two Single Input Single Output (SISO) examples in which a performance study of this proposed GPC software and a comparison with the PID controller are done. The last section is dedicated to conclude this paper.

II. Theoretical background: review of the GPC

In this section, the direct output method described in state space model developed by [16] is adopted to design the GPC controller. The dynamic system is described by the following discrete state-space model:

$$x(k+1) = Fx(k) + G\Delta u(k) \tag{1a}$$

$$y(k) = Hx(k) \tag{1b}$$

in which $x(k) \in \mathbb{R}^n$ is the state of the system, $\Delta u(k)$ is the input, $y(k)$ is the measured output and the operator $\Delta = 1 - z^{-1}$ denotes the integral action which ensures static error elimination.

Without loss of generality, we assume, for simplicity, that the state space matrices are in the observer canonical form as follows:

$$F = \begin{bmatrix} -a_1 & I & 0 & \cdots & 0 \\ -a_2 & 0 & I & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ -a_{n-1} & 0 & 0 & \cdots & I \\ -a_n & 0 & 0 & \cdots & 0 \end{bmatrix}, \quad G = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{n-1} \\ b_n \end{bmatrix} \tag{2a}$$

$$H = \overbrace{\begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}}^{n} \tag{2b}$$

where : $b_i = 0 \quad \text{for} \quad i > n_b$.

Consequently, we can obtain using equation 1 the following state at time $k + j$:

$$x(k+j|k) = F^j x(k) + \sum_{i=0}^{j-1} F^{j-1-i} G\Delta u(k+i) \tag{3}$$

Furthermore, by using equations 1 and 3, the j-step ahead output predictor value is written as follows:

$$\widehat{y}(k+j|k) = HF^j x(k) + \sum_{i=0}^{j-1} HF^{j-1-i} G\Delta u(k+i) \tag{4}$$

An usual form of the performance criterion used to compute the control law is a quadratic one given by:

$$J = \frac{1}{2}[\sum_{j=1}^{H_p} (\widehat{y}(k+j|k) - w(k+j))^2 + \lambda \sum_{j=1}^{H_c} \Delta u(k+j-1)^2] \tag{5}$$

where $\widehat{y}(k+j|k)$ is given by equation 4 and $w(k+j)$ denotes the set-point at time $k + j$.

It is easier to use the matrix form. Therefore, the output sequence on H_p prediction horizon can be written as follows:

$$Y = L\Delta U + Mx(k) \tag{6}$$

in which:

$Y = [\widehat{y}(k+1|k), \widehat{y}(k+2|k), \ldots, \widehat{y}(k+H_p|k)]^T$,

$\Delta U = [\Delta u(k), \Delta u(k+1), \ldots, \Delta u(k+H_c-1)]^T$,

It is assumed that there is no control action after time $k + H_c - 1$, i.e. $\Delta u(k+i) = 0$ for $i > H_c - 1$. Since the GPC is a receding horizon approach, only the first computed control move $\Delta u(k)$ is implemented.

The L matrix with the (H_p, H_c) dimension and M which is an (H_p, n) dimensional matrix are given by:

$$L = \begin{bmatrix} HG & 0\ldots & 0 & \\ HFG & HG & 0 & \\ \vdots & \vdots & \ddots & \\ HF^{H_p-1}G & HF^{H_p-2}G & \ldots & HFG \end{bmatrix},$$

$$M = \begin{pmatrix} HF \\ HF^2 \\ \vdots \\ HF^{H_p} \end{pmatrix}.$$

The objective function can be expressed as:

$$J = \frac{1}{2}[(Y-W)^T(Y-W) + \lambda \Delta U^T \Delta U] \tag{7}$$

in which:

$$W = [w(k+1), \ldots, w(k+H_p)]^T$$

Substituting 6 into 7 and minimizing it with respect to U gives:

$$\Delta U = [L^T L + \lambda I_{Hc}]^{-1} L^T [W - Mx(k)] \tag{8}$$

where $I_{H_c} \in \mathbb{R}^{H_c \times H_c}$ is the identity matrix.

Finally, we obtain the following recursive form of the control input:

$$u(k) = u(k-1) + \widetilde{L}[W - Mx(k)] \tag{9}$$

in which: $\widetilde{L}$ denotes the first row of $[L^T L + \lambda I_{Hc}]^{-1} L^T$.

III. STM32 starter kit and Keil development tool

In this section, an overview of the hardware and the software development tools is presented.

A. Why STM32F103RB microcontroller

First of all ,we have to clarify why the choice of a microcontroller? Most of the proposed works of the implementation of the predictive control method use only the FPGA or the PLC. May be this is due to the complexity of use of the microcontroller and the huge amount of time needed to implement such method above all when incorporating constraints. However, by proposing an optimized algorithm with a reducing size of code and by choosing a low cost microcontroller with a low power

consumption, one can takes advantages of such miniaturized device.

Moreover, the choice to adopt STM32 is based on trade offs between price (since the STM32 discovery has a very low price), performance, and low power consumption. Indeed, the STM32F103RB performance line family incorporates the high-performance ARM Cortex-M3 32-bit RISC core operating at a maximum of 72 MHz frequency, high-speed embedded memories (Flash memory of 128 Kbytes size and SRAM of 20 Kbytes size), and an extensive range of enhanced I/Os and peripherals connected to two APB buses. All devices offer two 12-bit ADCs, three general purpose 16-bit timers plus one PWM timer, as well as standard and advanced communication interfaces: two I2Cs and SPIs, three Universal synchronous asynchronous receiver transmitter USARTs, an USB and a CAN. Moreover, the STM32F103RB has configurable and flexible power management features that allow to choose the power option to fit application. You can dynamically manage the power consumption or hardware to match the system's requirements. Power management is provided via clock control to the CPU and individual peripherals. This device supports the following three global power control modes: The STM32F10xxx devices feature three low-power modes:

- Sleep mode (CPU clock off, all peripherals including Cortex-M3 core peripherals like NVIC, SysTick, etc. are kept running),
- Stop mode (all clocks are stopped),
- Standby mode (1.8V domain powered-off).

In addition, the power consumption in Run mode can be reduced by one of the following means:

- Slowing down the system clocks,
- Gating the clocks to the APB and AHB peripherals when they are unused.

This is obviously an attractive property to industry as saving power and CPU high frequency can be a very costly affair indeed. The block diagram of the STM32F103RB microcontroller is presented by the Fig. 1.

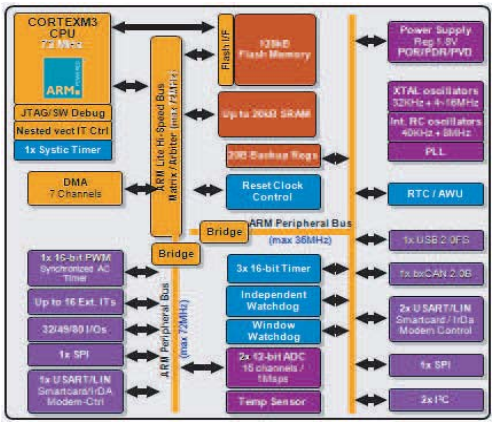


Fig. 1. STM32F103RB block diagram

B. STM32 starter kit

The STM32 starter kit presented in Fig. 2 was used to implement the GPC program.

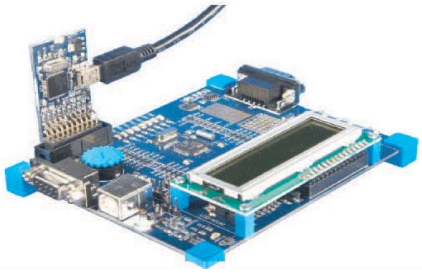


Fig. 2. STM32 starter kit

It is composed of:

- MCBSTM32 Evaluation Board which includes:
  - STM32F103 with 72 MHz maximum Cortex-M3 processor based MCU with 128KB Flash, 20KB RAM, CAN, USB, 2 x 12-bit 16 channel ADC's, and 49 GPIO.
  - Serial Port, CAN, USB Interfaces, and SD/MMC card slot.
  - 16x2 LCD panel, 8 LED's, 3 push buttons, GPIO, and scratchpad area.
- A JTAG interface supporting Cortex-M3 Serial Wire Debugger (SWD) and Serial Wire Viewer (SWV) modes.

In order to load the program into the STM32 device, the ULINK-ME is used.

C. Keil development tools

Keil is a software developement tools. It makes C/C++ compilers, debuggers, integrated environments, middleware, real-time kernels, simulation models, and evaluation boards for ARM, Cortex-M, Cortex-R4, 8051, C166, and 251 processor families. The used version is the μVision 4. The μVision 4 screen provides a menu bar for command entry, a tool bar where can select command buttons, and windows for source files, dialog boxes, and information displays. μVision 4 can simultaneously open and view multiple source files. This version has two operating modes:

- *Build Mode:* Allows to translate all the application files and to generate executable programs. The features of the Build Mode are described under Creating Applications.
- *Debug Mode:* Provides a powerful debugger for testing your application. The Debug Mode is described in Testing Programs.

In both operating modes you may use the source editor of μVision 4 to modify the source code. The Debug mode adds additional windows and stores an own screen layout. Moreover, this tool has the ability to communicate information to the serial port of the PC monitor.

IV. Firmware Development

In the proposed algorithm, the Generalized Predictive Control is implemented (see Fig. 3).

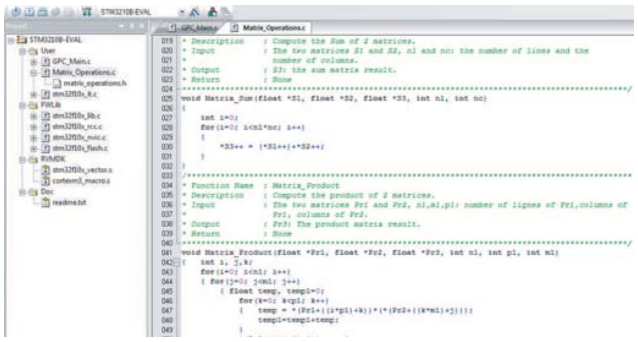


Fig. 3. Structure of GPC algorithm in Keil μVision4 development tool

A. Software description of the GPC controller

This subsection describes the GPC controller software and gives details about the related functions. It is important to notice that automatically generating matlab algorithms into C/C++ environment is time demanding, therefore in this work we choose to implement all the predictive control algorithm in C code. In spite of its complexity, this method will be more optimized.

In order to use the GPC controller software, some steps should be followed:

- define the number of states, the number of iterations, the prediction horizon, the control horizon and the weighting factor λ,
- define the state matrix, the input matrix and the output matrix of the model,
- fix the initial values of the control and output vectors.

Below, we present the different functions developed in this firmware, their description, their inputs and their outputs which allow finally to compute the optimal control action.

- GPC_Main: it presents the main program of the GPC controller. A call of routines of the matrix operations is made when needed. In order to be executed, this main program need the declaration of the model, the prediction horizon, the control horizon, the weighting factor and the number of iterations. It requires also the initialization of the control signal.
- Matrix_Operations: it contains all the matrix manipulation functions described below:

1) Zero_Matrix: it is described in Table I and allows to initialize all the matrices.

TABLE I
ZERO_MATRIX FUNCTION DESCRIPTION

Function name	Zero_Matrix
Function prototype	void Zero_matrix(float *T, int n_z, int m_z)
Behavior description	Matrix initialization
Input	T: the matrix to be transposed, n_z: the number of lines of T, m_z: the number of columns of T.
Output	None

2) Matrix_Product: it is described in Table II and allows the multiplication of two matrices.
3) The Matrix_Sum: it is described in Table III.

TABLE II
MATRIX_PRODUCT FUNCTION DESCRIPTION

Function name	Matrix_Product
Function prototype	void Matrix_Product(float *P_1, float *P_2, float *P_3, int n_1, int p_1, int m_1)
Behavior description	Compute the product of two matrices
Input	P_1: the first left matrix, P_2: the second right matrix, P_2: the product matrix, n_1: the number of lines of P_1, p_1: the number of lines of P_2, m_1: the number of columns of P_2.
Output	None

TABLE III
MATRIX_SUM FUNCTION DESCRIPTION

Function name	Matrix_Sum
Function prototype	void Matrix_Sum(float *S_1, float *S_2, float *S_3, int n_l, int n_c)
Behavior description	Compute the sum of two matrices
Input	S_1: the first matrix, S_2: the second matrix, S_3: the sum matrix , n_l: the number of lines of S_1, n_c: the number of columns of S_1.
Output	None

TABLE IV
MATRIX_TRANS FUNCTION DESCRIPTION

Function name	Matrix_Trans
Function prototype	void Matrix_Transp(float* M, float* M_t, int n , int m)
Behavior description	Compute the transpose of a matrix
Input	M: the matrix, M_t: the transpose matrix, n: the number of lines of M, m: the number of columns of M.
Output	None

4) The Matrix_Trans: it is described in Table IV.
5) The Prod_Vect_Sca : it is described in Table V.

TABLE V
PROD_VECT_SCA FUNCTION DESCRIPTION

Function name	Prod Prod_Vect_Sca
Function prototype	void Prod_Vect_Sca(float *V, float *S, float *VS, int n_{11}, int m_{11})
Behavior description	Compute the product of a vector with a scalar
Input	V: the vector, S: the scalar, VS: the result , n_{11}: the number of lines of V, m_{11}: the number of columns of V.
Output	None

6) Matrix_inverse : This routine returns the resulting inverted matrix using augmented matrix with the Gauss Jordan algorithm. Its parameters are described in table VI.

V. SIMULATION EXAMPLES

The main features of the proposed GPC controller software are illustrated through two examples. All these examples were tested in run mode and executed from the flash memory. In order to allow the implementation of such a computationally expensive controller on chip, we propose reducing the frequency of the STM32 CPU to 24MHz while maintaining good

TABLE VI
MATRIX_INVERSE FUNCTION DESCRIPTION

Function name	Matrix_Inverse
Function prototype	void Matrix_Inverse(float *M, float *Inver, int nlines)
Behavior description	Compute the inverse of a matrix
Input	M: the matrix, $Inver$: the Inverted matrix, nlines: the number of lines of columns of M.
Output	None

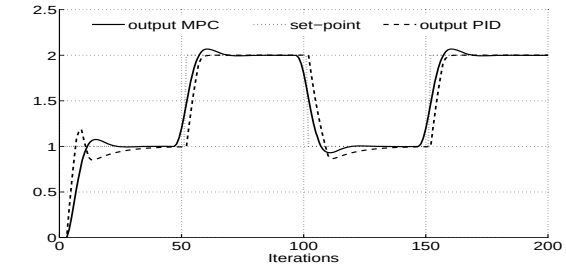
performance. This value is the maximum frequency of STM32 discovery which is characterized by a very low price and additional peripherals comparing with the proposed STM32 Keil board (for example: it contains a DAC: Digital to Analog Converter).

A. A First order plant: Comapraison with the antiwindup PID controller

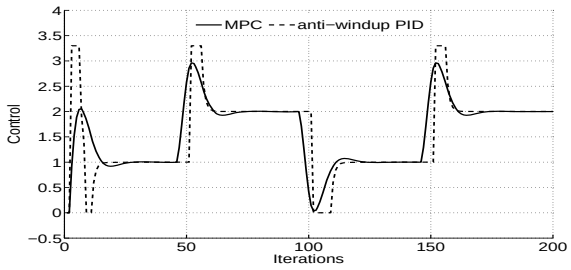
The first example is a simple first order system which has the following discrete time model:

$$y(k) = \frac{0.09516z^{-1}}{1 - 0.9048z^{-1}}\Delta u(k)$$

The control parameters of the GPC method are: $H_p = 5$, $H_c = 1$ and $\lambda = 1$. The method used to implement the anti-windup PID controller, in the STM32 using Keil tool, is the takahashi method anti saturation of the integral term. The PID controller parameters found using the closed loop takahashi method are: $K_p = 5.7$, $K_i = 0.1$ and $K_d = 0.25$. Moreover, the PID control constraints are : $u_{max} = 3.2$ and $u_{min} = 0$.



(a)



(b)

Fig. 4. Closed-loop simulation results for 1st order system

Although basing on table VII, the execution time of the on-line algorithm of the PID is less than that of the MPC, the last method has more parameters such as λ, H_c and H_p which allows controlling the performances of the closed loop response. Indeed, from the simulation results of figure Fig. 4, we notice that the MPC has the advantage of predicting the behavior of the output with respect to changes of the set-point.

TABLE VII
PERFORMANCE COMPARISON BETWEEN PID AND MPC CONTROLLER

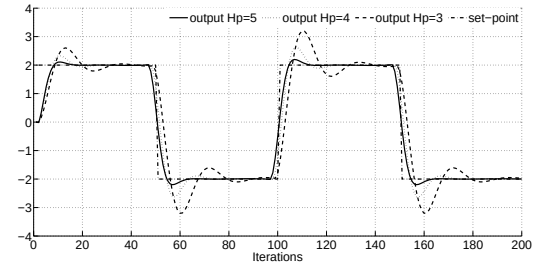
	PID anti wind up controller	MPC controller
The average execution time per sample (μs)	3.5	26.518
The flash memory code size (KBytes)	7.472	5.888

B. A second order plant

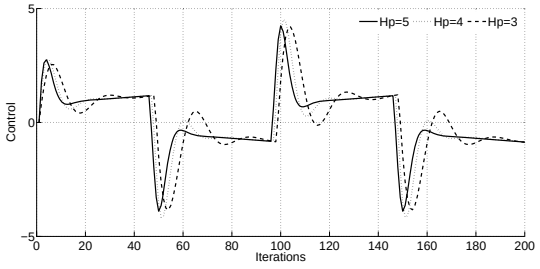
The proposed process is a speed control which consists in a motor fitted with a speed sensor, the control objective is to regulate the speed of the motor by manipulation of the input voltage. The mathematical model of the system with a sampling time of 0.1 sec is [17]:

$$\Delta x(k+1) = \begin{bmatrix} 0.93 & -0.01 \\ 0.04752 & 0.9964 \end{bmatrix} \Delta x(k) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \Delta u(k)$$
$$\Delta y(k) = \begin{bmatrix} -0.01 & 3.71 \end{bmatrix} \Delta x(k)$$

The figure 5 presents the closed loop results with the following controller parameters : the control horizon is $H_c = 2$ and the weighting factor $\lambda = 1$ and for different values of H_p.



(a)



(b)

Fig. 5. Closed-loop simulation results for speed control process

VI. Interpretations and results

Fig 5 shows the closed loop simulation results for speed control process for different prediction horizon. Based on these results, we deduce that the proposed software GPC contoller based on STM32 successfully controls the above systems with a good set-point tracking.

Table VIII lists the performance of the proposed software for different prediction horizon and for $H_c = 2$.

TABLE VIII
Performance study for Hc=2

The prediction horizon Hp	3	4	5
The average execution time per sample (μs)	37.286	43.464	49.643
The execution time of the off line algorithm (μs)	313	523.125	788.542
The flash memory code size (KBytes)	6.056		

Table IX lists the performance of the proposed software for different prediction horizon and for $H_c = 1$.

TABLE IX
Performance study for Hc=1

The prediction horizon Hp	3	4	5
The average execution time per sample (μs)	35.536	41.321	47.107
The execution time of the off line algorithm (μs)	225.666	376.667	565.167
The flash memory code size (KBytes)	6.056		

Based on these tables, it can be seen that when the prediction horizon or the control horizon grow higher, the computation time of the on-line MPC algorithm increases slightly.

VII. Conclusion

A framework for embedding SISO model predictive control on a performed STM32 microcontroller has been provided. The STM32 Keil starter kit was used to implement the GPC controller firmware. Two examples were used to test the MPC STM32 firmware. Hence, an efficient implementation of this code yields a low computational burden with a high speed. Indeed based on the simulation results, we notice that the proposed software controls successfully the processes with a good set-point tracking. Moreover, the PID controller has been implemented in the same microcontroller and compared with the proposed GPC software. All these results should allow MPC to be used in application areas where the computational load has been considered too great until now. There are still much detailed analysis and tests to be done, which should consider the constraints in the process and handle multivariable systems.

References

[1] T. Alamo, D. Ramirez, and E. Camacho, "Efficient implementation of constrained min-max model predictive control with bounded uncertainties: a vertex rejection approach," *Journal of Process Control*, vol. 15, pp. 149–158, 2005.
[2] E. Camacho and C. Bordons, *Model Predictive Control*. London: Springer, 2004.
[3] G. D. Nicolao, L. Magni, and R. Scattolini, "Robust predictive control of systems with uncertain impulse response," *Automatica*, vol. 32, no. 10, pp. 1475–1479, 1996.
[4] P. Campo and M. Morari, "Robust model predictive control," in *Proceedings of American Control Conference*, 1987, pp. 1021–1026.
[5] J. Rossiter, *Model based predictive control: A practical approach*. Robert H.Bishop, 2005.
[6] M. Kothare, V. Balakrishnan, and M. Morari, "Robust constrained model predictive control using linear matrix inequalities," *Automatica*, vol. 32, no. 10, pp. 1361–1379, 1996.
[7] Y. Wang and J. Rawlings, "A new robust model predictive control method 1: theory and computation," *Journal of Process Control*, vol. 14, pp. 231–247, 2002.
[8] G. Pannochia, "Robust model predictive control with guaranteed set point tracking," *Journal of Process Control*, vol. 14, pp. 927–937, 2004.
[9] B. Bouzouita, F. Bouani, and M. Ksouri, "Solving non convex min-max predictive controller," in *Proceedings of Information, Decision and Control Conference*, 2007.
[10] A. Kheriji, F. Bounani, and M. Ksouri, "Efficient implementation of constraind robust model predictive control using a state space model," in *Proceedings of International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, 2010.
[11] A. Kheriji, F. Bouani, and M. Ksouri, "Ggp approach to solve non convex min-max robust model predictive controller for a class constrained mimo systems," in *Proceedings of the International Workshop on Symbolic and Numerical Methods, Modeling and applications to circuit design, (SM2ACD) CEDA competition*, 2010.
[12] A. Kheriji, F. Bounani, and M. Ksouri, "A ggp approach to solve non convex min-max predictive controller for a class of constrained mimo systems described by state-space models," *International Journal of Control Automations and Systems (IJCAS)*, vol. 9, no. 3, 2011, will appear in June.
[13] K. Ling, S. Yue, and J. Maciejowski, "A fpga implementation of model predictive control," in *Proceedings of American Control Conference*, 2006.
[14] U. R. Y. Jayaraman, "Fpga implementation of predictive control strategy for power factor correction," in *Proceedings of World Academy of Science, Engineering and Technology*, 2008.
[15] K. Ling, B. Wu, and J. Maciejowski, "Embedded model predictive control (mpc) using a fpga," in *Proceedings of the 17th World Congress: The International Federation of Automatic Control, Seoul, Korea*, 2008.
[16] K. Watanabet, K. Ikeda, T. Fukuda, and S. Tzafestas, "Adaptive generalized predictive control using a state space approach," in *International Workshop on Intelligent Robots and Systems IROS, Osaka, Japan*, 1991.
[17] G. Palomo, K. Hilton, and J. Rossiter, "Predictive control implementation in a plc using the iec 1131.3 programming standard," in *Proceedings of American Control Conference*, 2009.

Amira Abbes Kheriji Amira Kheriji Abbes was born in Tunis, Tunisia, in 1982. She received the Eng. Degree in Electrical Engineering and the Master degree in the Automatic and Signal Processing from the National School of Engineering of Tunis (ENIT), in 2006 and 2008 respectively. From 2006 to 2008, she worked in STMicroelectronics of Tunis as application and support engineer in the microcontroller division and she was responsible of the ST microcontroller low power aspect. She is currently preparing her Ph.D in the laboratory of Analysis and Control of Systems at ENIT. Her main research interests include constrained predictive control, robust predictive control, real time systems, implementation of the control algorithms.

Faouzi Bouani Faouzi Bouani is a professor at the National School of Engineering of Tunis (ENIT). He received his B.Sc. and M. Sc. degrees, respectively in 1990 and 1992, from the Ecole Normale Suprieure de l'Enseignement Technique of Tunis. He received his Ph. D. degree in 1997 and the Habilitation universitaire in 2007, in Electrical Engineering both from (ENIT). His research interests include nonlinear predictive control, robust predictive control, and the computational intelligence technique.

Mekki Ksouri received the M.A. degree in physics in the FST in Tunis in 1973, the M.E. degree Ingenieur SupElec (Ecole Suprieure d'Electricite) in Paris 1973, the D.Sc. degree, and the Ph.D. degree from the University of Pierre and Marie Curie, Paris, France, in 1975 and 1977 respectively. He is Professor at the National School of Engineering of Tunis (ENIT). He was principal of the National Institute of Applied Sciences and Technology (INSAT) from 1999 to 2005, principal and founder of the High School of Statistics and Information Analysis from 2001 to 2005 and the High Institute of Technologic Studies from 1996 to 1999 and principal of The High Normal School of Technological Education from 1978 to 1990. He is also the author of 6 books. His activities focus on automatic control, robust control, and optimization in planning and scheduling, including implementation of fuzzy logic, neural nets, and genetic algorithms. He had received the Outstanding Contribution Award for Leadership in the Organization of the CIFA 2004 Symposium, Chevalier dans l'Ordre des Palmes Acadmiques France 2002, Outstanding Contribution Award IEEE SMC 1998. He is a senior member IEEE and he was the head and founder of many scientific associations (ATTNA, ASET,...).