Predictive Control for Relative Performance Management

Dharma Aryani
Department of Electrical Engineering
State Polytechnic of Ujung Pandang
Makassar, Indonesia
dharma.aryani@poliupg.ac.id

Nur Asyik Hidayatullah *Department of Electrical Engineering State Polytechnic of Madiun* Madiun, Indonesia
asyik@pnm.ac.id

***Abstract*—This paper examines the implementation of Model Predictive Control (MPC) for relative performance management in a shared resources system. Finite Control Set MPC as a model-based control is integrated in a scheme of differentiated control for a multi classes virtualized software system. A dynamic model is estimated in block-oriented form with nonlinear compensation feedback system. The performance objective is to maintain response time on the reference levels or subject to the degree of priority between the user classes. Experiments are conducted in a two-classes of virtualized software system for several performance differentiation scenarios. The performance of Integral FCS-MPC is evaluated in comparison with Proportional Integral (PI) control. The results show that predictive control framework provides significant improvement in predictability and disturbance rejection procedure. Therefore, Integral FCS-MPC outperforms PI controller in terms of maintaining the stability of relative performance objectives.**

***Index Terms*—Model Predictive Control, nonlinear compensation, Finite Set Control, Hammerstein-Wiener, virtual machines control, relative performance control**

I. INTRODUCTION

The dynamic characteristic of a shared resources environment comprises unpredictable conditions in application workload and performance objectives. A self-adapting capability will address these challenges by means of dynamic resources management in relative configuration. However, the essential control problem in relative management is to guarantee the achievement of performance preferences by managing the constraints of resources allocation between users. In addition, control system has to deal with nonlinear dynamics of such shared resources system. Therefore, studies about the effective mechanism to manage relative performance management and its nonlinearity issue have reached high attention.

Dynamic resource management regulates the resources to share them proportionally between users with regard to the performance references. This approach brings benefits for resource sharing and performance properties management [1], [2]. Existing studies proposed dynamic resource allocation based on control engineering and non-control engineering methods. Non-control engineering approaches utilize simple rule-based methods [3] or complex optimization techniques [4]. However, they have limitations in design parameters, and lack of systematic processes to achieve system stability [5]. In contrast, the control engineering methods provide a systematic design process, and capability to deal with model uncertainty. Furthermore, feedback principles have the ability to cope with unpredictable changes in operating conditions [6], [7]. On top of that, promising findings of feedback control approach by [8], [9] have shown a system identification with nonlinear compensation model to eliminate the issue of dynamic nonlinearities.

Finite Control Set MPC (FCS-MPC) is a class of MPC which has been proven for its algorithm simplicity to set input operating point as the control action. Earlier works by [10], [11] have shown that FCS-MPC is able to deal with dynamic management challenges. FCS-MPC is one of the most well-known predictive techniques for power converters and drives control ([12]). In addition, FCS-MPC is an an effective control with direct application that covers a wide range of objectives in power electronics applications. A brief review of recent applications of FCS-MPC in power electronics is addressed in [13].

This paper contributes to the implementation of Integral Finite Control Set MPC for relative performance management. Integral FCS-MPC optimizes the system performance by manipulating finite control points with an addition of integral action. Experiments are accomplished in a two-classes virtualized software system with alternative differentiation scenarios. In comparison with Proportional and Integral (PI) controller, the results exhibit the robustness of Integral FCS-MPC with better system stability in complex conditions of resources constraints and objective preferences.

The presentation in this paper is divided into five sections. Section II covers the description of virtualized software system and its model structure. Section III presents the general formulation of Integral FCS-MPC and PI controller. The experimentation and results analyses are provided in Section IV, followed by a brief summary in Section V.

II. SYSTEM DESCRIPTION

A. Virtualized Software System

An isolated network of virtual machines is established to represent a shared-resources system. Figure 1 shows the testbed architecture that comprises of server, database and client workload simulator. $RUBiS$ model is implemented as the software system application to realize an auction site

978-1-5386-5529-0/18/$31.00 ©2018 IEEE

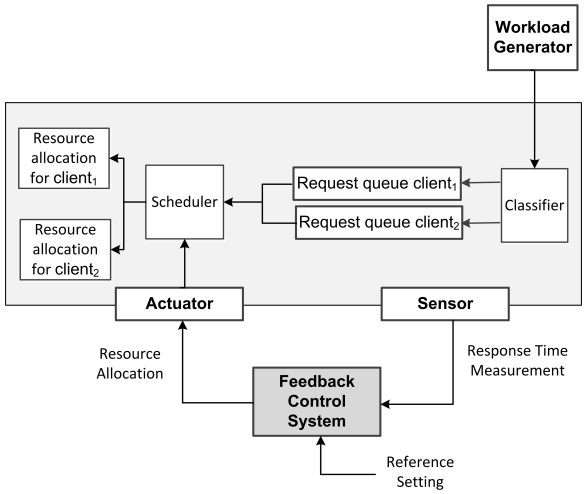


Fig. 1: Abstraction of control loop architecture in two classes shared resource software system

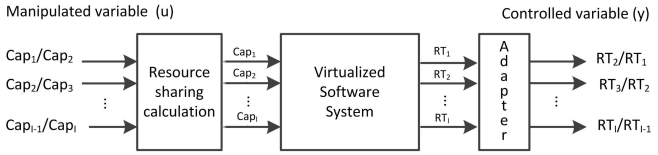


Fig. 2: Structure of relative management scheme

benchmark for multi-tiers network. The server infrastructures are shared among two virtual machines (VMs) that use the resources correspondingly. Server takes the role as host machine to serve the guest/client machines which are two virtualized machines. $client_1$ and $client_2$ represent VM_1 and VM_2, respectively. An actuator feeds the required ratio of CPU allocation into the system, while a sensor program is enclosed to each VM for the response time measurement of requests workload. The virtualization procedure is executed using $Xen2.6$ hypervisor with a default scheduler for proportional CPU provisioning for each VM.

Relative scheme of performance management designates a relative significance of clients and maintain the ratio of performance metrics to the desired values. In other words, the CPU allocation is dynamically regulated to assure the reference value of response time can be achieved. The structure of relative management and objectives for I clients can be seen in Fig. 2. At sampling time k, input variable is $u(k) = \frac{Cap_1(k)}{Cap_2(k)}$ which provides CPU capacity entitlement for VM_1 over VM_2. The share of resource allocation is in the percentage of total CPU capacity that equals to 100%. Output response is the ratio of measured time for the host to respond the client requests, $y(k) = \frac{RT_2(k)}{RT_1(k)}$. $RT_1(k)$ and $RT_2(k)$ represent response time to the workloads from VM_1 and VM_2 respectively. The reference value for output $y(k)$ is determined by the preference or priority level of each client (P_1 and P_2), the ratio value is $r(k) = \frac{P_2(k)}{P_1(k)}$. Increasing the resource provisioning for a VM leads to a faster response time to its workloads.

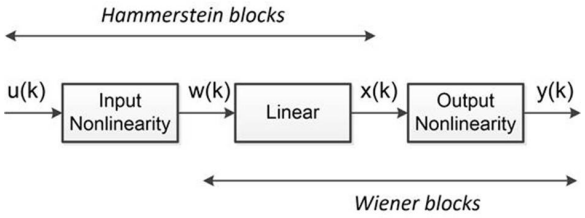


Fig. 3: Hammerstein-Wiener structure

B. System Identification

Dynamics of virtualized software system are characterized in block-oriented system identification. Model estimations refer to the previous finding in [9] using a Hammerstein-Wiener model structure. Fig.3 shows the model blocks which consist of two nonlinear memoryless blocks and a linear dynamic block. The nonlinear relationship of input signal u and intermediate input w is captured within the Hammerstein block in an inverse static nonlinearity formulation. In Wiener block, a linear model characterizes the linear dynamics followed by an inverse function which approximates the nonlinearity in B-spline function. The estimated inverse nonlinear models are integrated as compensator for the nonlinear dynamics in both input and output elements of feedback system. The inverse static input nonlinearity model is in a function $u(k) = f^{-1}(w(k))$ are approximated by setting $w(k)$ values with $w_{min} = -15$, $w_{max} = 15$ and $\delta w = 0.5$. A polynomial function of $u(k)$ is derived by using leats squares estimation.

$$\begin{aligned} u(k) = {} & 4.17e^{-7}w(k)^5 + 9.34e^{-6}w(k)^4 + \\ & 1.02e^{-4}w(k)^3 + 0.003w(k)^2 + \\ & 0.08w(k) + 1.005 \end{aligned} \tag{1}$$

The nonlinear characteristic is estimated in an inverse static nonlinear model. Input variable is $y(k)$ and output is $x(k)$ which is the intermediate output variable. Thus, the representative formulation of B-spline curve $B(y(k))$ is written as follows,

$$x(k) = \sum_{i=0}^{s} N_{i,\rho}((y(k))Cp_i \tag{2}$$

where N_i is basis function,Cp_i is control points of the curve, and ρ is the curve order. The linear model ($w(k) = f(x(k))$) is an ARX model;

$$x(k+1) = 0.4201x(k) + 0.0430w(k) \tag{3}$$

III. Finite Control Set - Model Predictive Control

FCS-MPC exploits a finite number of input states for solving the optimization function. Prediction for system response uses the discrete model of system dynamics where all the possible state combinations are evaluated at each sampling time. The output are compared to the reference setting by assessing a cost function (J). Consequently, a state with minimum J value is selected as the optimal control input.

Basic algorithm of FCS is a receding horizon approach with one-step ahead estimation and on-line optimization. The value of $y(k)$ for estimation uses the updated value from current measurements of output response.

Discrete model of a dynamic system is defined as,

$$\begin{bmatrix} y_1(k+1) \\ y_2(k+1) \end{bmatrix} = A \begin{bmatrix} y_1(k) \\ y_2(k \end{bmatrix} + B \begin{bmatrix} u_1(k) \\ u_2(k) \end{bmatrix} \tag{4}$$

The cost function which is solved in least squares optimization is expressed below,

$$\begin{aligned} J = \begin{bmatrix} r_1(k) - y_1(k+1) & r_2(k) - y_2(k+1) \end{bmatrix} \cdot \\ \begin{bmatrix} r_1(k) - y_1(k+1) \\ r_2(k) - y_2(k+1) \end{bmatrix} \end{aligned} \tag{5}$$

A. Integral FCS-MPC

Optimal FCS performance will degrade in complex system constraints because of the steady-state error. Therefore, the Finite Control Set method is revised with the addition of integral action in the controller, this will be denoted as Integral FCS (IFCS). The basic objective of a finite control set is to achieve an optimal output of feedback control by deploying a gain matrix K_{fcs}. The optimal control signal $u(k)$ is written as,

$$\begin{bmatrix} u_1(k)^{opt} \\ u_2(k)^{opt} \end{bmatrix} = K_{fcs} \left(\begin{bmatrix} r_1(k) \\ r_2(k) \end{bmatrix} - \begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix} \right) \tag{6}$$

where K_{fcs} denotes the feedback gain for the reference value and the measured output response. The FCS controller gain (K_{fcs}) is calculated from the equation below,

$$K_{fcs} = B^{-1} A \tag{7}$$

The integrated error signal from reference value and measured output response are inserted into controller. An integrator in discrete-time system is formulated as $k_I \dfrac{1}{1 - z^{-1}}$, where z^{-1} is the backward shift operator to represent $z^{-1}x(k) = x(k-1)$. Consequently, this term is added to the basic equation of FCS that is called Integral FCS-MPC. The revised form is expressed as,

$$\begin{aligned} \begin{bmatrix} u_1(k)^{opt} \\ u_2(k)^{opt} \end{bmatrix} = K_{fcs} \begin{bmatrix} \frac{k_I}{1-z^{-1}}(r_1(k) - y_1(k)) \\ \frac{k_I}{1-z^{-1}}(r_2(k) - y_2(k)) \end{bmatrix} \\ - K_{fcs} \begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix} \end{aligned} \tag{8}$$

where k_I is the integral gain for the input signal. The value of the integral gain is in a range of $0 < k_I \leq 1$. The actual control signal $u(k)$ for IFCS is chosen by evaluating the objective function for all the available control signals candidates. The $u(k)$ value is the optimal solution of the one-step ahead prediction algorithm in FCS.

$$J = (u_1(k) - u_1(k)^{opt})^2 + (u_2(k) - u_2(k)^{opt})^2 \tag{9}$$

Fig. 4 illustrates the Integral FCS-MPC structure.

The configuration comprises an inner-loop for proportional control and an outer-loop for integral control. The design of integral controller determines the closed-loop transfer function.

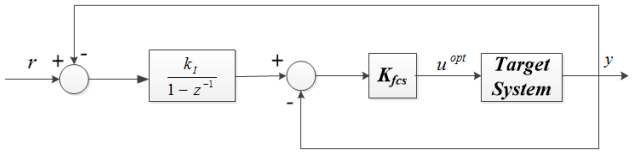


Fig. 4: Integral FCS-MPC feedback structure

Inner-loop has a proportional gain K_{fcs} and outer-loop has an integral gain k_I. The inner-loop feedback controller provides relationship of $u(k)$ and $y(k)$,

$$y(k+1) = ay(k) + bu(k) \tag{10}$$

Control variable $u(k)$ is calculated as

$$u(k) = K_{fcs}(e(k) - y(k))$$

where $e(k)$ is the control signal of the outer-loop, and also the reference value for the inner-loop. Afterwards, the outer-loop feedback system which has an integral action, can be designed in a straightforward manner. The outer-loop includes a function of integral action $\dfrac{k_I}{1 - z^{-1}}$ and a time delay z^{-1} from inner-loop. Thus, the transfer function of outer-loop system is formulated as

$$\frac{Y(z)}{R(z)} = \frac{k_I z^{-1}}{1 - z^{-1} + k_I z^{-1}} \tag{11}$$

where the closed-loop pole is selected to be $1 - k_I$. Selection of closed-loop pole in a range of $0 \leq p_{cl} \leq 1$ leads to an integral gain $k_I = 1 - p_{cl}$. The pole p_{cl} is a design parameter that is selected and predefined by user. Hence, closed-loop pole location can be adjusted according to the desired time constant of feedback system. With $\Delta y = y(k) - y(k-1)$, integral gain leads to a formulation of $u(k)^{opt}$ which is written below,

$$\begin{aligned} u(k)^{opt} = u(k-1)^{opt} + K_{fcs}(K_i(y^*(k) - y(k))) \\ - K_{fcs}(\Delta y(k)) \end{aligned} \tag{12}$$

B. Proportional Integral Control

The linear model of target system is represented in transfer function $G(z)$ and the controller in $C(z)$. There are four main variables in the control loop, i.e. reference value (R_z) , control input (U_z), measured output (Y_z) and control error (E_z). The transfer function in a first order ARX model is

$$G(z) = \frac{Y(z)}{U(z)} = \frac{b_1}{z + a_1} \tag{13}$$

Proportional control is used when the control signal is assigned to be proportional with the change of process error value $e(z) = r(z) - y(z)$. The gain of proportional controller is denoted as K_p. Integral control overcomes the drawbacks of proportional control by reducing or eliminating the steady offset without significantly increasing the controller gain. The

integral gain is denoted as K_i. The transfer function for PI controller is formulated below,

$$C(z) = \frac{U(z)}{E(z)} = K_p + \frac{K_i}{1 - z^{-1}} = \frac{(K_i + K_p)z - K_p}{z - 1} \tag{14}$$

Controller gains (K_p and K_i) are obtained using pole placement method. It is an approach for tuning PI gains by specifying the desired closed-loop poles location. The feedback control loop is ,

$$\frac{Y(z)}{R(z)} = \frac{C(z)G(z)}{1 + C(z)G(z)} \tag{15}$$

The denominator polynomial is also known as characteristic equation of the control system. Therefore, the roots of polynomial are equal to the desired closed-loop poles. If polynomial function of the chosen closed-loop is,

$$D_{cp} = z^2 - (cp_1 + cp_2)z + cp_1 cp_2 \tag{16}$$

The gains are derived by equating (15) and (16).

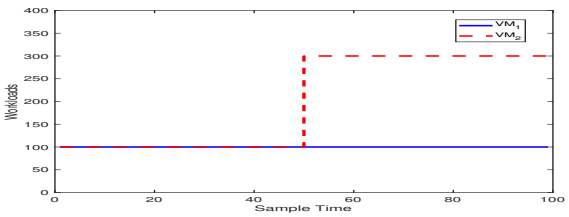
$$K_p = \frac{cp_1 cp_2 + a_1}{-b_1} \tag{17}$$

$$K_i = \frac{(cp_1 + cp_2) + 1 - a_1}{b_1} - K_p \tag{18}$$

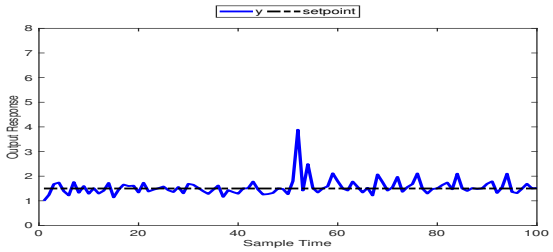
IV. Experimentation and Analysis

Control system is integrated in a feedback control structure with nonlinear compensation technique. The compensator function at input and output will reduce the effect of nonlinear dynamics to the control system performance. Integral FCS-MPC and PI controller are evaluated in three scenarios which have been formulated to ensure constraint and disturbance variation in operational objectives during runtime. Controller gains are calculated with regards to the linear model in Eq. 3 by setting the desired pole values. The selection of pole location is determined after analyzing response stability spesifications, such as settling time, steady state error, overshooting, and the flow rate of control signal. Pole positions imposing varied behaviour of control system because the occurrence of system nonlinearities. Thus, simulations have been conducted in order to find the best values of poles, cp_1 and cp_2 are at 0.5. Using the pole location, control gains are calculated based on Eq. 17 and 18 which leads to the value of $K_p = 0.86$ and $K_i = 0.13$. Furthermore,controller gains for Integral FCS-MPC are determined from equations in Section III-A that yields to $K_{fcs} = 9.76$ and $K_i = 0.5$.

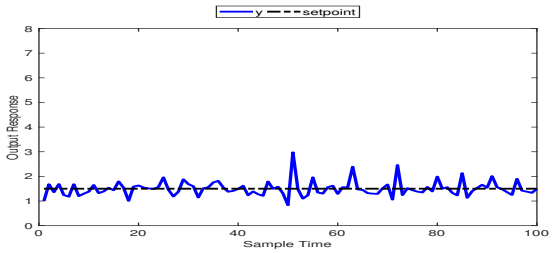
Scenario A : Reference $>$ 1 *:* In this experiment, the priority levels between both VMs is set to $\frac{P_2}{P_1} = 1.5$. This reference value means that the requests from $client_1$ get the first priority, which means they should be responded faster than the requests from $client_2$. Therefore, $client_1$ will be given more resources than $client_2$ during runtime. Fig. 5a shows the workload settings for experiment in scenario A. It represents a scenario in which higher workloads are suddenly imposed



(a) Workloads disturbance in scenario A



(b) Output response PI-FC



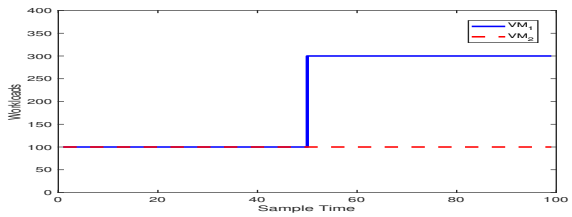
(c) Output response IFCS

Fig. 5: Experimental results of PI-based feedback control and Integral FCS-MPC in scenario A

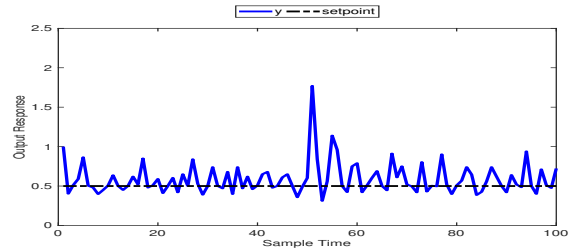
the virtual machine that has lower priority in accessing the resources. Therefore, a robust control for CPU provisioning is needed to sustain the preferred output response. Results in Fig. 5b-5c show the performance of both control approaches. The statistics summary of Mean Square Error (MSE), minimum and maximum value of output responses is presented in Table I.

Scenario B : Reference $<$ 1*:* This setting illustrates that $Client_2$ is more important than $client_1$. The priority levels between both VMs is set to $\frac{P_2}{P_1} = 0.5$. This value represents the condition where requests from $client_2$ need a faster response than requests from $client_1$. Therefore, the system should serve $client_2$ as the first priority, with a bigger ratio of resource allocation. Fig. 6 shows workload settings and measured output of each control system. The steady state error evaluation is summarized in Table I.

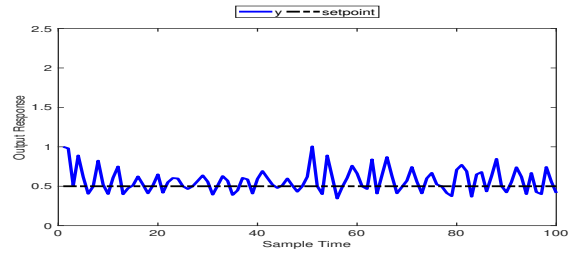
Scenario C : Reference changes during runtime: In this scenario, the level of importance of $client_1$ and $client_2$ is alternately changed. This case shows the adaptation ability to adjust resources according to the desired performance objectives. The setpoint is maintained at 1 until the $50th$ sample. Then, after the $50th$ sample, it is increased to 1.5. Workload in all sampling times is 200 and 300 requests/sec



(a) Workloads disturbance in scenario B

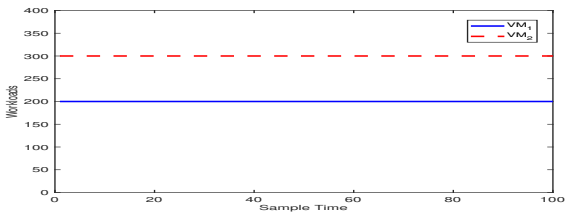


(b) Output response PI-FC

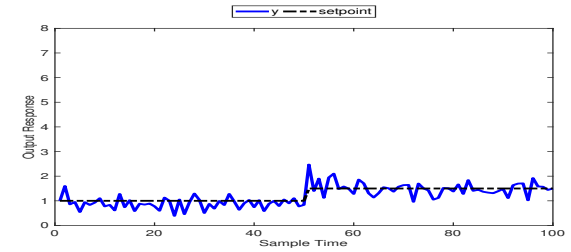


(c) Output response IFCS

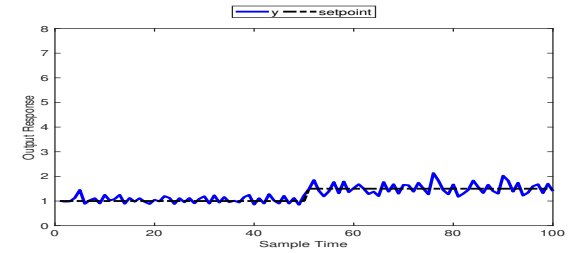
Fig. 6: Experimental results of PI-based feedback control and Integral FCS-MPC in scenario B



(a) Workloads disturbance in scenario C



(b) Output response PI-FC



(c) Output response IFCS

Fig. 7: Experimental results of PI-based feedback control and Integral FCS-MPC in scenario C

TABLE I: Statistical summary of steady state response in all experiment settings

	A		B		C	
	PI	IFCS	PI	IFCS	PI	IFCS
MSE	0.11	0.09	0.04	0.03	0.03	0.01
Min	0.83	0.24	0.13	0.34	0.39	0.85
Max	3.88	2.97	1.77	1.01	2.47	2.13

for VM_1 and VM_2 respectively. This is a scenario in which both virtual machines have a workload demand bigger than the nominal value. Priority level of the VM with higher workload is reduced in the middle of runtime. This change causes the VM with higher demand to be granted lower priority access to resources. The workload setting and output responses are captured in Fig. 7. The experiment results and statistics have shown that model-based predictive control in a form of Integral FCS provides more robust performance management for virtualized software system compare to PI-based feedback control. The MSE values of IFCS control system are maintained smaller than the PI in all experiments. When reference signals were placed in sensitive region, disturbances were rejected efficiently under high workloads. For underload and overload conditions in scenario A and B, the response time output of PI system contains larger steady state error than IFCS control. This analysis can be defined from the maximum and minimum value of output response. Further evaluation is based on the overshoot of signal response in all scenarios. A higher overshoots of transient response exhibit a lower adaptability of control system to deal with unpredictable changes of operating system.

Moreover, it can be seen from the output responses in scenario C that the controllers can adapt responsively to the changes in reference value of priority level. Accordingly, it is clear that Integral FCS-MPC algorithm improves the predictability and disturbance rejection capability of control system. Another considerable factor that improves the control performance is the novel approach for model estimation which contributes to nonlinear compensation of system dynamic. These findings will complement the previous studies for performance management that have implemented basic MPC [8] and subspace-based MPC [14].

V. CONCLUSIONS

The Integral Finite Control Set MPC has been studied and implemented for relative performance management and resource provisioning in a shared resources system. Based on the output response from complex scenarios of experimentation,

it can be inferred that feedback control system with IFCS demonstrates more robust response in any changes of reference value and workload disturbances. IFCS control system yields to smaller steady state error than the response from PI-based controller. All performance improvements are caused by the prediction nature of MPC framework with an addition of integral action. Therefore, in any scenario of disturbance and service differentiations, the system will adaptively respond without sacrificing the steady state condition.

REFERENCES

[1] J. L. Hellerstein, Y. Diao, S. Parekh, and D. M. Tilbury, *Feedback control of computing systems*. John Wiley & Sons, 2004.

[2] M. Karlsson, X. Zhu, and C. Karamanolis, "An adaptive optimal controller for non-intrusive performance differentiation in computing services," in *Control and Automation, 2005. ICCA'05. International Conference on*, vol. 2. IEEE, 2005.

[3] L. Chenyang, Y. Lu, T. F. Abdelzaher, J. A. Stankovic, and S. Sang Hyuk, "Feedback Control Architecture and Design Methodology for Service Delay Guarantees in Web Servers," *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 9, 2006.

[4] V. Hien Nguyen, F. D. Tran, and J. M. Menaud, "Performance and Power Management for Cloud Infrastructures," in *The 3rd IEEE International Conference on Cloud Computing (CLOUD)*, 2010.

[5] T. Patikirikorala, A. Colman, J. Han, and L. Wang, "A systematic survey on the design of self-adaptive software systems using control engineering approaches," in *Proceedings of the 7th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*. IEEE Press, 2012.

[6] C. Lu, T. F. Abdelzaber, J. A. Stankovic, and S. H. Son, "A feedback control approach for guaranteeing relative delays in web servers," in *Real-Time Technology and Applications Symposium, 2001. Proceedings. Seventh IEEE*. IEEE, 2001, pp. 51–62.

[7] P. Padala, K. G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, A. Merchant, and K. Salem, "Adaptive control of virtualized resources in utility computing environments," in *ACM SIGOPS Operating Systems Review*, vol. 41, no. 3. ACM, 2007, pp. 289–302.

[8] T. Patikirikorala, L. Wang, A. Colman, and J. Han, "Hammerstein–Wiener nonlinear model based predictive control for relative QoS performance and resource management of software systems," *Control Engineering Practice*, vol. 20, no. 1, 2012.

[9] D. Aryani, L. Wang, and T. Patikirikorala, "On identification of hammerstein and wiener model with application to virtualised software system," *International Journal of Systems Science*, vol. 48, no. 6, pp. 1146–1161, 2016.

[10] R. Vargas, U. Ammann, and J. Rodriguez, "Predictive approach to increase efficiency and reduce switching losses on matrix converters," *IEEE Transactions on Power Electronics*, vol. 24, no. 4, pp. 894–902, 2009.

[11] D. E. Quevedo, R. P. Aguilera, M. A. Perez, P. Cortés, and R. Lizana, "Model predictive control of an afe rectifier with dynamic references," *IEEE Transactions on Power Electronics*, vol. 27, no. 7, pp. 3128–3136, 2012.

[12] S. Kouro, P. Cortés, R. Vargas, U. Ammann, and J. Rodríguez, "Model predictive control—a simple and powerful method to control power converters," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 6, pp. 1826–1838, 2009.

[13] J. Rodriguez, M. P. Kazmierkowski, J. R. Espinoza, P. Zanchetta, H. Abu-Rub, H. A. Young, and C. A. Rojas, "State of the art of finite control set model predictive control in power electronics," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 2, pp. 1003–1016, 2013.

[14] D. Aryani, L. Wang, and T. Patikirikorala, "Data-driven predictive control with nonlinear compensation for performance management in virtualized software system," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 7795–7800, 2017.