PERGAMON

Automatica 39 (2003) 35–45

automatica

www.elsevier.com/locate/automatica

Run-to-run control methods based on the DHOBE algorithm☆

Chang Zhang, Hao Deng, John S. Baras*

Department of Electrical and Computer Engineering, Institute for Systems Research, University of Maryland, College Park, MD, 20742, USA

Received 22 March 2000; received in revised form 20 August 2001; accepted 30 August 2002

Abstract

Since process models are typically not known exactly in real problems, it is important to estimate the process parameters before one applies the optimal control to a process. In this paper, the Dasgupta–Huang optimal bounding ellipsoid (DHOBE) algorithm is employed to estimate process parameters in semiconductor process run-to-run (RtR) control. At each iteration, the DHOBE algorithm returns an outer bounding ellipsoid of the likely process parameter set. If the vector center of the ellipsoid is taken as the estimate of the process parameter vector, then a model-reference controller results; if the vector within the ellipsoid that produces the worst expected cost is taken as the process parameter estimate, then a worst-case controller results. These two methods are compared with other RtR control schemes: the exponentially weighted moving average (EWMA) method and the optimizing adaptive quality controller (OAQC). Simulation results show that the performance of the model-reference RtR controller based on the DHOBE algorithm is comparable to or better than that of the other two RtR controllers in some specific examples of semiconductor processes.

Keywords: Run-to-run control; DHOBE algorithm; EWMA algorithm; OAQC; Set-valued method; OVE algorithm

1. Introduction

In industries such as semiconductor manufacturing, limitations and costs impose a need for adjusting process parameters on a run-to-run (RtR) basis. This need has originated a collection of techniques called RtR control (Baras & Patel, 1996; Sachs, Hu, & Ingolfsson, 1995; Boning, Moyne, & Smith, 1995; Castillo & Yeh, 1998; Hankinson, Vincent, Irani, & Khargonekar, 1997; Ingolfsson & Sachs, 1993; Ning, 1996; Deng, 1999). RtR control is a form of discrete-time process control in which the product recipe with respect to a particular equipment process is modified ex situ, i.e. between equipment "runs", so as to compensate for the effects of process drifts, large shifts, and other disturbances to keep the outputs at the prescribed target values. A drift disturbance, which may be produced by the equipment aging, change of environment or other factors, causes a slow and constant change of the outputs of a process. Different from a drift disturbance, a shift disturbance (step disturbance) causes a large change of the outputs of a process in a few runs. It may be produced by the failure of a component, change of the operator or other reasons.

Generally, a RtR controller is designed in the following way. First, it computes an optimal control based on the initial process model. The initial process model can be derived from former off-line experiments such as using the response surface model (RSM) method. The controller does not modify the recipe during a run. At the end of the current run, the controller updates the process model based on the new measurements. Finally, at the beginning of the next run, it adjusts the recipe according to the updated process model. A typical block diagram of a RtR controller is illustrated in Fig. 1. The reason why it generates the new recipe from the post-process measurements on a run-to-run basis is, on one hand, lack of on-line sensors for the process. On the other hand, frequent changes of inputs to the process may increase the variability of the process's output (Sachs et al., 1995). Sometimes, a deadband is utilized in order to make the model changes less frequent.

☆ This paper was not presented at any IFAC meeting. This paper was recommended for publication in revised form by Associate Editor Peter Fleming under the direction of Editor Sigurd Skogestad. Work partially supported by NSF under grant ECS 972 7805.

* Corresponding author. Fax: +1-301-314-9218.

E-mail addresses: zchang@isr.umd.edu (C. Zhang), hdeng@bechtel.com (H. Deng), baras@isr.umd.edu (J.S. Baras).

0005-1098/03/$ - see front matter © 2002 Elsevier Science Ltd. All rights reserved.
PII: S0005-1098(02)00197-8

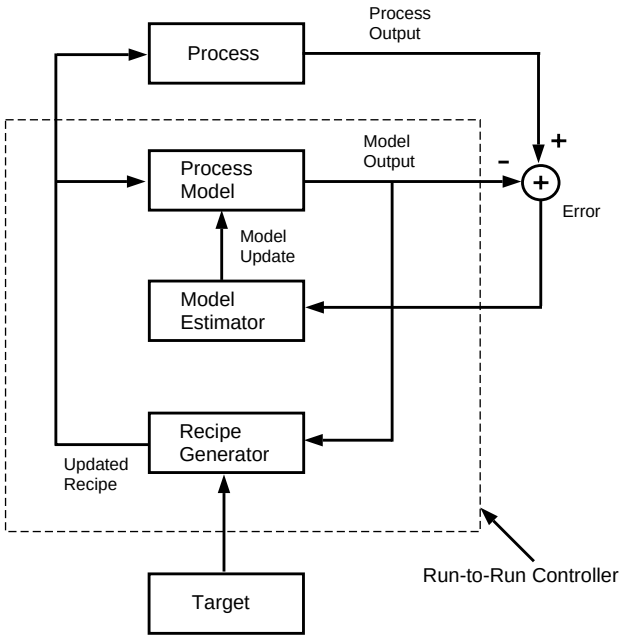


Fig. 1. Structure of a RtR controller.

Currently, there are three main RtR control schemes available:[1] The exponentially weighted moving average (EWMA) method, the least-squares estimation (LSE) method and the set-valued RtR controller.

1. The EWMA method (Sachs et al., 1995) is widely used in RtR control for its simplicity and efficiency to compensate for smooth drifts and other small disturbances. The EWMA method uses a linear (affine) model to approximate a process and only updates the offset term in the model. There are many modifications to the EWMA controller. For example, a deadband may be added to the EWMA controller to further reduce the variation of the process output (Sachs et al., 1995). Statistical methods can be combined with the EWMA controller to identify the existence of a shift and estimate the shift's magnitude. Then a rapid mode can be applied by adjusting the model parameters quickly (Sachs et al., 1995). The weight of the EWMA controller is an important factor that affects its performance. In (Smith & Boning, 1997), the authors use a neural network to adjust the weight of the EWMA controller. The neural network has to be trained off-line before it is deployed. A double exponential forecasting filter, which has two EWMA modules, is also developed to predict and remove drifts in a process (Butler & Stefani, 1994).

2. The LSE method. Typical examples are the optimizing adaptive quality controller (OAQC) (Castillo & Yeh, 1998) and the Kalman filter based approach (Palmer, Ren, & Spanos, 1996). The OAQC uses a second-order model to approximate a process. It may have better performance than the EWMA controller in controlling a non-linear process. All the coefficients of the model in the OAQC can be adjusted from run to run (Castillo & Yeh, 1998). The Kalman filter based approach uses a linear model to describe a process. Different from an EWMA controller, the Kalman filter based method can adaptively adjust both the slope and the intercept terms (Palmmer et al., 1996). Therefore, the LSE method based RtR controllers may have stronger tracking ability than the EWMA controller.

3. The set-valued RtR control method (Baras & Patel, 1996). Due to measurement errors and environment noises, it is difficult to find an exact process model. The location of the likely process model parameters for the next optimization run form a set. We could be quite certain that the parameter vector is somewhere in this set. Due to disturbances, the exact model parameters are unknown. An outer-bounding ellipsoid is usually used to approximate the set of likely parameter values; the ellipsoid is used for its simplicity. In (Baras & Patel, 1996), they apply the optimal volume ellipsoid (OVE) algorithm (Cheung, Yurkovich, & Passino, 1991) to find the bounding ellipsoid. In this paper, we are going to apply a more general ellipsoid algorithm, the Dasgupta–Huang optimal bounded ellipsoid (DHOBE) algorithm (Dasgupta & Huang, 1987) to approximate the likely model parameter set. The DHOBE algorithm updates the parameter estimates only when the new measurements contain new information. This reduces significantly the computational load for the set-valued RtR control method to estimate the process model parameters. The DHOBE algorithm was improved in (Rao & Huang, 1993) by introducing a rescue procedure. When the process undergoes abrupt shifts and modeling errors, the rescue procedure improves the performance of the algorithm and accordingly that of the controller. Therefore, a DHOBE-algorithm-based controller may work well for large step disturbances and model errors, which may be hard for other control methods to compensate for. Moreover, the DHOBE-algorithm-based controller can use a nth-order polynomial model to approximate a process and adjust all the coefficients adaptively. Hence, it may even have stronger tracking performance than the LSE-method-based controllers.

At each iteration, the DHOBE algorithm returns an outer bounding ellipsoid that contains the true parameter vector with high probability. If the vector center of the ellipsoid is taken as the estimate of the parameter vector, the explicit model update is implemented and leads to a model-reference method. If we search for the worst expected output that may be produced by a vector within the ellipsoid and then minimize the worst-case cost, a set-valued worst-case controller is obtained.

The DHOBE algorithm and the design of the corresponding controllers will be introduced in Section 2. The DHOBE-algorithm-based controllers will be compared with

[1] There are also some other RtR control methods such as the probabilistic approach (Hamby, Kabamba, & Khargonekar, 1998) and the machine learning approach Ning, 1996. They are not so effective and will not be introduced in this paper.

the EWMA controller and the OAQC in Section 3. Section 4 includes our conclusions from our experiments/simulations.

2. The RtR controller based on the DHOBE algorithm

The DHOBE algorithm is employed to obtain the ellipsoid that bounds the likely process model parameter set.

2.1. The DHOBE algorithm

Assume that a multi-input and single-output process has the following model:

$$y_t = \theta_t^{*\mathrm{T}} u_t + \eta_t, \tag{1}$$

where $t \in \Re$ is the discrete time index, $t = 1, 2, \ldots$, $y_t \in \Re$ is the scalar output of the process, $\theta_t^* \in \Re^N$ is the true process parameter vector and $u_t \in U \subset \Re^N$ is the input vector. U is the control space. $\eta_t \in \Re$ is the noise term. Assume that η_t is bounded by a finite number $\gamma \in \Re^+$, i.e., $|\eta_t| \leqslant \gamma$. [2]

Suppose that at time instant $t-1$, the membership set of the process model parameters is bounded by the ellipsoid E_{t-1}. The ellipsoid is defined by its center $\theta_{t-1} \in \Re^N$, its orientation matrix $P_{t-1} \in \Re^{N\times N}$ and its size (the uncertainty parameter) $r_{t-1}^2 \in \Re^+$

$$E_{t-1} = \{\theta \in \Re^N : [\theta - \theta_{t-1}]^{\mathrm{T}} P_{t-1}^{-1} [\theta - \theta_{t-1}] \leqslant r_{t-1}^2\}. \tag{2}$$

At the next time instant t, we have available a new observation of the process output y_t. Then we can utilize it to obtain the set S_t in which the process parameters will reside under the constraint of the noise as follows:

$$S_t = \{\theta \in \Re^N : [y_t - \theta^{\mathrm{T}} u_t]^2 \leqslant \gamma^2\}. \tag{3}$$

S_t will intersect with E_{t-1}, which enables us to iteratively enclose the intersection of the two sets by a new ellipsoid E_t as follows (Rao & Huang, 1993):

$$\begin{aligned} E_t = \{&\theta \in \Re^N : (1-\lambda_t)[\theta - \theta_{t-1}]^{\mathrm{T}} P_{t-1}^{-1} [\theta - \theta_{t-1}] \\ &+ \lambda_t [y_t - \theta^{\mathrm{T}} u_t]^2 \\ &\leqslant (1-\lambda_t) r_{t-1}^2 + \lambda_t \gamma^2\}, \end{aligned} \tag{4}$$

where the updating gain λ_t ($0 \leqslant \lambda_t \leqslant 1$) is introduced. We can also consider $(1-\lambda_t)$ as the forgetting factor. λ_t is chosen so as to minimize r_t^2 in order to decrease the size of the ellipsoid from run to run. The iteration of the DHOBE algorithm is shown in Fig. 2(a). The DHOBE algorithm can be generalized as follows:

Step 1: Given an observed output y_t and the input vector u_t, compute the error residual δ_t by

$$\delta_t = y_t - u_t^{\mathrm{T}} \theta_{t-1}. \tag{5}$$

[2] In practice, the noise bound may be unknown. How to select the noise bound for a specific application will be introduced later in this paper.

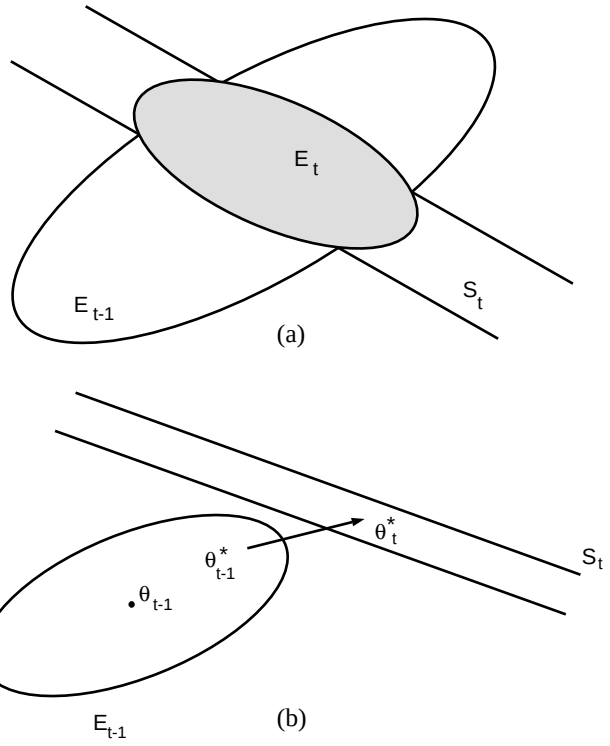


Fig. 2. (a) Iteration of the bounding ellipsoid, where E_t is found to include the intersection $E_{t-1} \cap S_t$ (Dasgupta & Huang, 1987). (b) The DHOBE algorithm rescue procedure. The size of E_{t-1} will be enlarged so that the intersection of S_t and the enlarged ellipsoid E_{t-1} is not empty. This rescue procedure will migrate the center of the ellipsoid towards the real parameter vector θ_t^*.

Step 2: If

$$r_{t-1}^2 + \delta_t^2 \leqslant \gamma^2, \tag{6}$$

then the noise is thought negligible and the bounding ellipsoid is not updated; otherwise, go to Step 3.

Step 3: Compute two intermediate scalar variables:

Step 3a:

$$G_t = u_t^{\mathrm{T}} P_{t-1} u_t. \tag{7}$$

Step 3b:

$$\beta_t = (\gamma^2 - r_{t-1}^2)/\delta_t^2. \tag{8}$$

Step 4: Compute the updating factor λ_t

$$\lambda_t = \min(\lambda_{\max}, \upsilon_t), \tag{9}$$

where

$$\upsilon_t = \begin{cases} \lambda_{\max} & \text{if } \delta_t^2 = 0, \\ (1-\beta_t)/2 & \text{if } G_t = 1, \\ \frac{1-\sqrt{G_t/1+\beta_k(G_t-1)}}{1-G_t} & \text{if } \beta_k(G_t-1) > -1, \\ \lambda_{\max} & \text{if } \beta_k(G_t-1) \leqslant -1 \end{cases} \tag{10}$$

and $\lambda_{\max}$ is a design parameter smaller than one, since $\lambda_t = 1$ implies that P_t will be singular (Dasgupta & Huang, 1987). This step returns a λ_t such that $r_t^2(\lambda_t) \leqslant r_t^2(\lambda)$ for all $\lambda \in [0, \lambda_{\max}]$, which means that the size r_t^2 of the ellipsoid E_t will be minimized by λ_t.

Step 5: Update the parameter uncertainty factor by (Dasgupta & Huang, 1987)

$$r_t^2 := (1-\lambda_t) r_{t-1}^2 + \lambda_t \gamma^2 - \frac{\lambda_t(1-\lambda_t)\delta_t^2}{1-\lambda_t+\lambda_t G_t}. \tag{11}$$

Step 6: This is the rescue procedure developed by Rao and Huang (1993). It is particularly necessary when the process parameters change abruptly in a few steps. In this case, the intersection of E_{t-1} and S_t may be empty as illustrated in Fig. 2(b). If the intersection is empty, then r_t^2 becomes negative which indicates that there is no bounding ellipsoid. At this moment, the rescue procedure is engaged to enlarge the size of E_{t-1} so that the intersection of S_t and the enlarged ellipsoid E_{t-1} is not empty. This rescue procedure will migrate the center of the ellipsoid towards the real parameter vector θ_t^*.

When $r_t^2 > 0$, proceed to Step 7; otherwise, compute

$$\kappa = \begin{cases} \delta_t^2 + \gamma^2 - 2\gamma|\delta_t| & \text{if } \lambda_t \neq \lambda_{\max}, \\ \lambda_{\max}\left[\dfrac{\delta_t^2}{1-\lambda_{\max}+\lambda_{\max}G_t} - \dfrac{\gamma^2}{1-\lambda_{\max}}\right] & \text{if } \lambda_t = \lambda_{\max}. \end{cases} \tag{12}$$

Reset the uncertainty parameter for time instant $t-1$

$$r_{t-1}^2 := \kappa + \zeta, \tag{13}$$

then return to Step 3b. Here ζ is called "inflation parameter". It is user-specified and usually set as 1 (McCarthy & Wells, 1997).

Step 7: Update the ellipsoid parameters

$$P_t := \frac{1}{1-\lambda_t}\left[P_{t-1} - \frac{\lambda_t P_{t-1} u_t u_t^{\mathrm{T}} P_{t-1}}{1-\lambda_t+\lambda_t G_t}\right] \tag{14}$$

$$\theta_t := \theta_{t-1} + \lambda_t P_t u_t \delta_t. \tag{15}$$

At the beginning of the DHOBE algorithm, we let $P_0 = I$ and give a large value to r_0 (e.g., $r_0^2 = 100$) to include the real model parameters. If there are l outputs in a process, then correspondingly there are l feasible parameter sets. Hence, l ellipsoids should be used to estimate the process parameters. By employing the DHOBE algorithm, we can design the model-reference RtR controller or the worst-case RtR controller.

2.2. The model-reference RtR controller based on the DHOBE algorithm

In the model-reference RtR controller, the vector center of the ellipsoid is used as the estimate of the process parameter vector. The recipe is generated by minimizing the squared error between the model's predicted outputs and the target values. We will call the resulting controller *DHOBE-MR*. The block diagram of the DHOBE-MR controller is shown in Fig. 3.

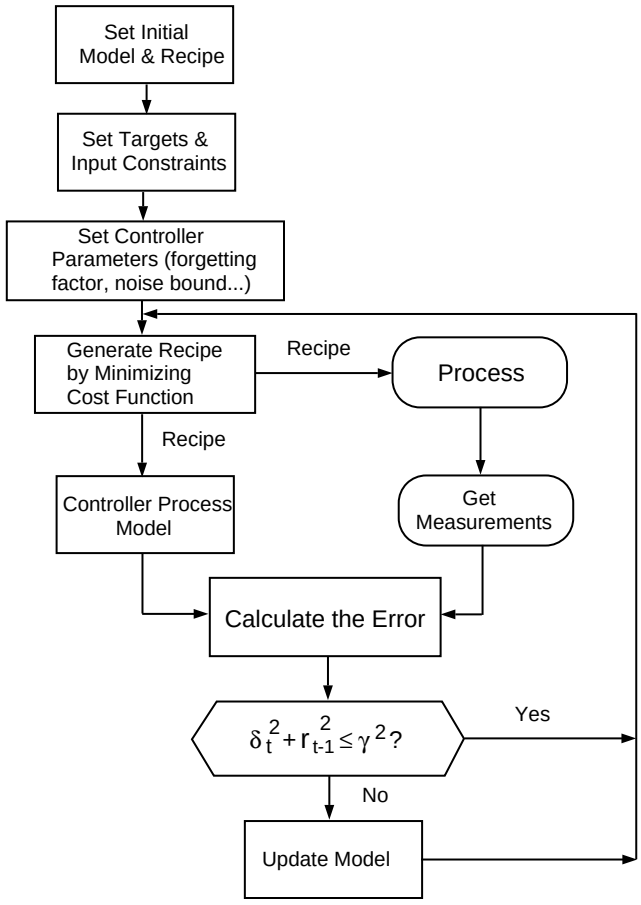


Fig. 3. Block diagram for the DHOBE algorithm based model-reference RtR controller.

At the beginning, the following parameters related to the internal process model need to be set:

- The initial model parameters, which might be obtained from off-line experiments.
- The target values of the process outputs.
- The constraints of the inputs.
- The initial recipe to minimize the cost function.

We also need to set the following controller parameters:

- The noise bound γ.
- The updating factor's upperbound $\lambda_{\max}$.
- The inflation parameter ζ (normally set to 1).
- The ellipsoid's orientation matrix P_0 at the beginning (normally set to unit matrix I).
- The ellipsoid's initial size r_0^2 that can be set to be a large number to contain the real parameters.

The noise bound and the upper bound of the updating factor can be obtained from off-line experiments and past experiences. From Step 2 of the DHOBE algorithm, it is easy to see that when the noise bound is large enough, the process model will not be updated until the process's output significantly deviates from the target. Large deviations are not desired in semiconductor manufacturing. If γ is chosen smaller than the actual bound, the bounding ellipsoid may not contain the real parameters and then the rescue procedure will be triggered. In our simulations, the process noise and measurement noise are combined into a noise with standard deviation σ_w. We take $3\sigma_w$ as the noise bound and it is found that this bound is appropriate enough for the simulated circumstances. The rescue step included in the DHOBE algorithm is intended to return the process parameters within the bounding ellipsoid, in the rare occasions when the disturbances place the parameters outside the bounding ellipsoid. The setting of the maximum value of the updating factor ($\lambda_{\max}$) is also important. It controls the rate of convergence of the algorithm. If it is very small, convergence of the algorithm is slow. If it is too large, the size of the ellipsoid (r^2) may change so rapidly that the ellipsoid's boundary excludes the real process parameter (θ_t^*). If θ_t^* ends up too far outside the ellipsoid, then the intersection of S_t and E_{t-1} is empty and the rescue procedure is invoked. The re-inflated ellipsoid may miss θ_t^* again at the next update cycle, resulting in a non-converging oscillation of the ellipsoid. In the experiments, we use 0.4 as the maximum value. The control results are usually satisfactory with the given parameters.

After obtaining the process model for the current run, the recipe can be calculated by minimizing a predefined cost function. For a multi-input single-output system, the cost function is given by $g(u) = (T - \theta_t^{\mathrm{T}} u)^2$, where $T \in \Re$ is the target, $\theta_t \in \Re^n$ is the parameter vector and $u \in \Re^n$ is the input vector. The recipe is obtained by minimizing the cost function

$$u_t = \underset{u \in U}{\operatorname{argmin}}\, g(u). \tag{16}$$

When there are multiple responses, the cost function may take the form

$$g(u) = (T - \theta_t^{\mathrm{T}} u)^{\mathrm{T}} W (T - \theta_t^{\mathrm{T}} u), \tag{17}$$

where $\theta_t \in \Re^{n \times l}$ is the parameter matrix, l is the dimension of the output vector, and $T \in \Re^l$ is the target value vector. W is a positive definite diagonal weight matrix. The value of each element in W represents the priority of each response. The most important output is assigned the largest weight. The recipe is again obtained by minimizing the cost function.

If the change of recipe incurs large costs, then this factor can also be taken into consideration:

$$\begin{aligned} g(u) = \{ & w_1 (T - \theta^{\mathrm{T}} u)^{\mathrm{T}} W (T - \theta^{\mathrm{T}} u) \\ & + w_2 (u - u_{t-1})^{\mathrm{T}} \Gamma (u - u_{t-1}) \}, \end{aligned} \tag{18}$$

where Γ is a positive definite diagonal weight matrix. If the cost for changing a certain input is high, then we can assign a large weight to that particular input. $w_1 > 0$ and $w_2 > 0$ are weight terms also. In general, the control space U is convex. Therefore, we are facing a standard convex constrained optimization problem (with the specification as above, this is a quadratic programming problem, which can be solved very fast by standard algorithms and commercially available software). Because our focus is on the process model estimation, the optimization part of the controller will not be introduced in detail in this paper.

After the inputs are adjusted accordingly, they are kept constant during the current run. At the end of the present run, we measure the output of the process. The residual error is calculated, and if necessary, the process model is updated.

2.3. The worst-case RtR controller based on the DHOBE algorithm

Different from the DHOBE-MR controller, the worst-case controller searches for the recipe to minimize the worst-case cost (Baras & Patel, 1996). Consider a MIMO system with n inputs and l outputs. Then we need l ellipsoids to estimate the models for the outputs. Suppose that the noise bound for the ith output is γ_i. At run t, the ith ellipsoid generated by the DHOBE algorithm has the vector center $\theta_{i,t}^*$, orientation $P_{i,t}$ and size $r_{i,t}^2$. The optimization problem is given by

$$\min_{u \in U} \max_{\underline{y}_t \leqslant y \leqslant \bar{y}_t} f(y), \tag{19}$$

where $f(y)$ is the cost function with respect to the output vector y, and $\bar{y}_t$ and $\underline{y}_t$ are the upper bound vector and lower bound vector, respectively. The elements of $\bar{y}_t$ and $\underline{y}_t$ can be obtained via

$$\underline{y}_{i,t} = \theta_{i,t}^{*\mathrm{T}} u - \sqrt{u^{\mathrm{T}} P_{i,t} r_{i,t}^2 u} - \gamma_i, \tag{20}$$

$$\bar{y}_{i,t} = \theta_{i,t}^{*\mathrm{T}} u + \sqrt{u^{\mathrm{T}} P_{i,t} r_{i,t}^2 u} + \gamma_i. \tag{21}$$

In order to simplify the calculation, we impose the restriction that the cost function $f(y)$ is convex with respect to y (e.g., typically we use as $f(y)$ a quadratic function of the deviation of the output vector y from the target value vector T). Therefore, the maximum cost will be achieved at one of the vertices of the box defined by $\bar{y}_t$ and $\underline{y}_t$ (Baras & Patel, 1996).

Compared with the model-reference approach, the worst-case approach is much more complex, since we have to solve a min–max problem. We will call the corresponding controller *DHOBE-SV*.

3. Simulations

In our simulations, we assume that the disturbances are unknown. The controllers' function is to adjust the process

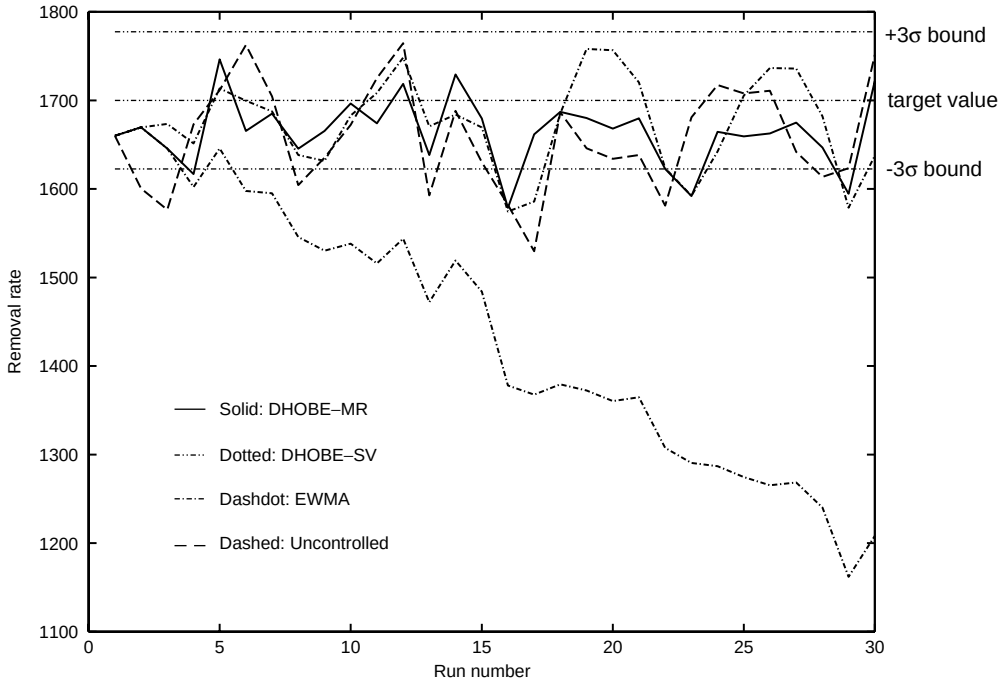


Fig. 4. Comparison of the EWMA controller, DHOBE-MR controller and the DHOBE-SV controller under drift and Gaussian noise. The response target value is fixed at 1700. The drift value is equal to -17 (unknown to the controllers) for each run and the variance of Gaussian noise is equal to 665.64 (unknown to the controllers).

models and compensate for those disturbances by post-measurements.

The DHOBE-algorithm-based controllers were compared with the EWMA controller (Sachs et al., 1995) and the OAQC (Castillo & Yeh, 1998). We performed 20 independent simulations for each scenario considered. Because our objective is to maintain the process outputs on targets, the main performance metric is $RMSD(y_i - T_i)$, the square root mean square deviation of the process's ith output from its target value.[3] The smaller its value, the better. In the rest of the paper, we will use the notation *RMSD* for simplicity.

3.1. Comparison of the DHOBE-MR and DHOBE-SV controllers with the EWMA controller

The underlying process model is obtained from a chemical mechanical planarization (CMP) process (Ning, 1996), where the units are dropped for simplicity:

$$y_t = -1382.60 + [50.18, -6.65, 163.4, 8.45]u_t^{\mathrm{T}} + w_t + \delta t, \quad (22)$$

where w_t is a normally distributed random variable with variance 665.64 and $\delta = -17$ is the drift value, which are assumed to be unknown to the controllers. The output target value is fixed at 1700. The controllers' objective is to maintain the output y_t as close to the target value as possible.

First, we assumed that we had perfect knowledge of the process model parameters at the beginning. The controllers were fully tuned to compensate for the disturbances based on post-measurements.[4] One of the 20 simulations is shown in Fig. 4. The weight of the EWMA controller in the figure is 0.6 (weight value was selected by optimizing performance over the weight range from 0.1 to 0.9, see the first row in Table 1(a)). The three straight lines are the $+3\sigma$, target and -3σ lines, respectively. The uncontrolled process diverges due to the drifts. All the controlled processes stay within the 3σ area most of the time. Because the weight of the EWMA controller plays an important role in its performance, we tested different weights for the EWMA controller. The *RMSD*s of the EWMA controller for different weights are listed in the first row of Table 1 (a). The *RMSD*s of the DHOBE-MR controller and the DHOBE-SV controller are listed in Table 1(b). From the table, one can see that the EWMA controller with the weights 0.6 and 0.7 gives slightly smaller *RMSD* than the DHOBE-MR controller for the unknown drift case. The *RMSD* of the DHOBE-SV controller is much larger than that of the DHOBE-MR controller in this case. The large *RMSD* of the DHOBE-SV controller may be due to the fact that this controller generates the recipe that

[3] In (Castillo & Yeh, 1998), the authors use the notation $MSD(y_i - T_i)$ for this performance metric.

[4] By adding an appropriate deadband to each controller, we may further reduce the *RMSD* produced by the controller. In our simulations, no deadband was used for anyone of the controllers.

Table 1
*RMSD*s of the EWMA controller with different weights under various disturbances. (b) *RMSD*s of the DHOBE-MR controller and the DHOBE-SV controller under various disturbances.

(a) *RMDs of the EWMA*

Weight	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Drift	135.39	82.03	61.48	52.62	47.03	45.38	45.39	49.95	67.78
Shift	322.55	214.33	172.94	152.50	141.85	136.46	140.31	155.31	168.09
Model error	164.52	134.80	122.81	124.41	130.97	148.46	191.55	364.00	1040.67

(b) *RMDs of the DHOBE-MR and DHOBE-SV controllers*

Noises	DHOBE-MR	DHOBE-SV
Drift	46.80	72.28
Shift	113.3	152.2
Model error	84.04	117.12

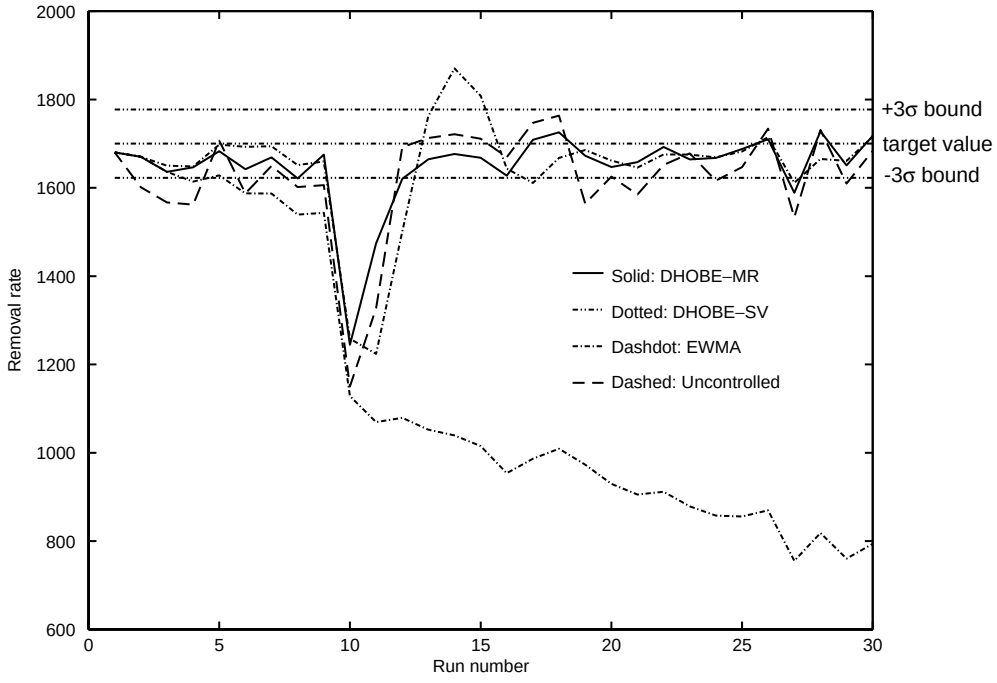


Fig. 5. Comparison of the EWMA controller, DHOBE-MR controller and the DHOBE-SV controller under shift, drift and Gaussian noise. The response target value is fixed at 1700. The drift value is equal to −17 (unknown to the controllers) for each run and the variance of Gaussian noise is equal to 665.64 (unknown to the controllers). The step disturbance occurs at run 10 by changing the values of the 'real' process parameters.

minimizes the worst-case cost, whilst this worst case rarely happens in practice. In other words, this controller is very pessimistic or very conservative.

Next, a shift disturbance was added to the underlying process (i.e., the disturbances included unknown drift, unknown shift and unknown Gaussian noise). The occurrence of the shift and its magnitude were unknown to the controller a priori. One of the 20 simulation runs is shown in Fig. 5. The weight of the EWMA controller selected in this figure is 0.6, based on the *RMSD* values in the second row of Table 1(a), which clearly show this weight to be the best performing weight. The *RMSD*s of the DHOBE-MR and DHOBE-SV controllers are provided in Table 1(b). From these experimental data, one can see that the DHOBE-MR controller gives the smallest *RMSD* among the three controllers in this case. The EWMA controller with weights 0.5, 0.6 and 0.7 results in smaller *RMSD*s than the DHOBE-SV controller. [5]

In real life, the underlying process model is unknown. To address this model uncertainty, in the following simulations we performed, the initial process model parameters used by the controllers were set at 80% of the true parameter values of the underlying process. This large model error

[5] One may be able to improve the performance of the EWMA controller by adding a rapid mode. This mode requires statistical methods to detect the magnitude of the shift and adjust the process model rapidly. Because there was no standard procedure on how to adjust the model, we did not apply the rapid mode in the experiments.

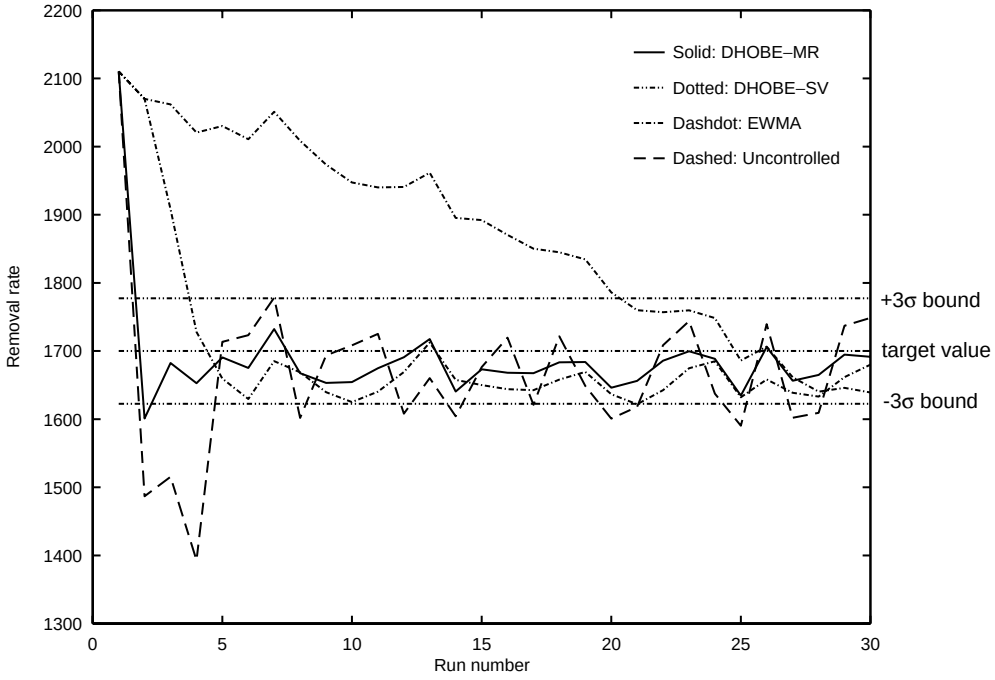


Fig. 6. Comparison of the EWMA controller, DHOBE-MR controller and the DHOBE-SV controller under unknown model error, drift and Gaussian noise. The response target value is fixed at 1700. The drift value is equal to −17 (unknown to the controllers) for each run and the variance of Gaussian noise is equal to 665.64 (unknown to the controllers).

should cause the output of the process to change abruptly at the beginning of the simulation. This was indeed observed, as shown in Fig. 6. The weight of the EWMA controller is 0.3 in this figure based on the performance evaluation results in the third row of Table 1(a). The disturbances included unknown model error, unknown drift and unknown Gaussian noise. The experimental performance data for the DHOBE-MR and DHOBE-SV controllers are shown in Table 1(b). One can see that the DHOBE-MR controller results in the smallest *RMSD* in the model-error case and the DHOBE-SV controller also gives smaller *RMSD* than the EWMA controller with all possible weights we tested.

In summary, when there are large model errors or large step disturbances, the DHOBE-MR controller may perform better than the EWMA controller with the best performing weight. The *RMSD* of the DHOBE-MR controller is 17% smaller in the shift case or 32% smaller in the model-error case than that of the EWMA controller with the best weight we tested. When the disturbances are small drifts and/or other small noises, the EWMA controller with a proper weight may perform slightly better than the DHOBE-MR controller. In the examples we run, we observed this in two out of nine experiments in the drift case, and the difference was less than 3%. The DHOBE-MR controller generally outperforms the DHOBE-SV controller, which is more conservative in its adjustment of the recipe. Therefore, when applying the DHOBE algorithm, we recommend the use of the model-reference version of the controller.

3.2. Comparison of the DHOBE-MR controller with the OAQC

Detailed descriptions of the OAQC can be found in (Castillo & Yeh, 1998). To make the comparison between the DHOBE-MR controller and the OAQC fair, we used the exact experimental conditions as described in (Castillo & Yeh, 1998). [6]

(1) *Process model considered as "real"*. The underlying "real" process is given by (Castillo & Yeh, 1998):

$$y_1 = 1563.5 + 159.3u_1 - 38.2u_2 + 178.9u_3 + 24.9u_4 - 67.2u_1u_2 - 46.2u_1^2 - 19.2u_2^2 - 28.9u_3^2 - 12u_1t' + 116u_4t' - 50.4t' + 20.4t'^2 + \varepsilon_{1,t}, \quad (23)$$

$$y_2 = 254 + 32.6u_1 + 113.2u_2 + 32.6u_3 + 37.1u_4 - 36.8u_1u_2 + 57.3u_4t' - 2.42t' + \varepsilon_{2,t}, \quad (24)$$

where $t' = (t - 53)/53$, $\varepsilon_{1,t} \sim N(0, 60^2)$, $\varepsilon_{2,t} \sim N(0, 30^2)$, and y_1 is the removal rate; its target value is 2000, y_2 is the with-in wafer non-uniformity; its target value is 100, u_1 is the platen speed, u_2 is the back pressure, u_3 is the polishing downforce, u_4 is the profile.

[6] Because the parameters of the OAQC are unknown, we could not replicate the OAQC results of (Castillo & Yeh, 1998) in our simulations. Instead, the simulation data of the OAQC performance were used as appeared in (Castillo & Yeh, 1998).

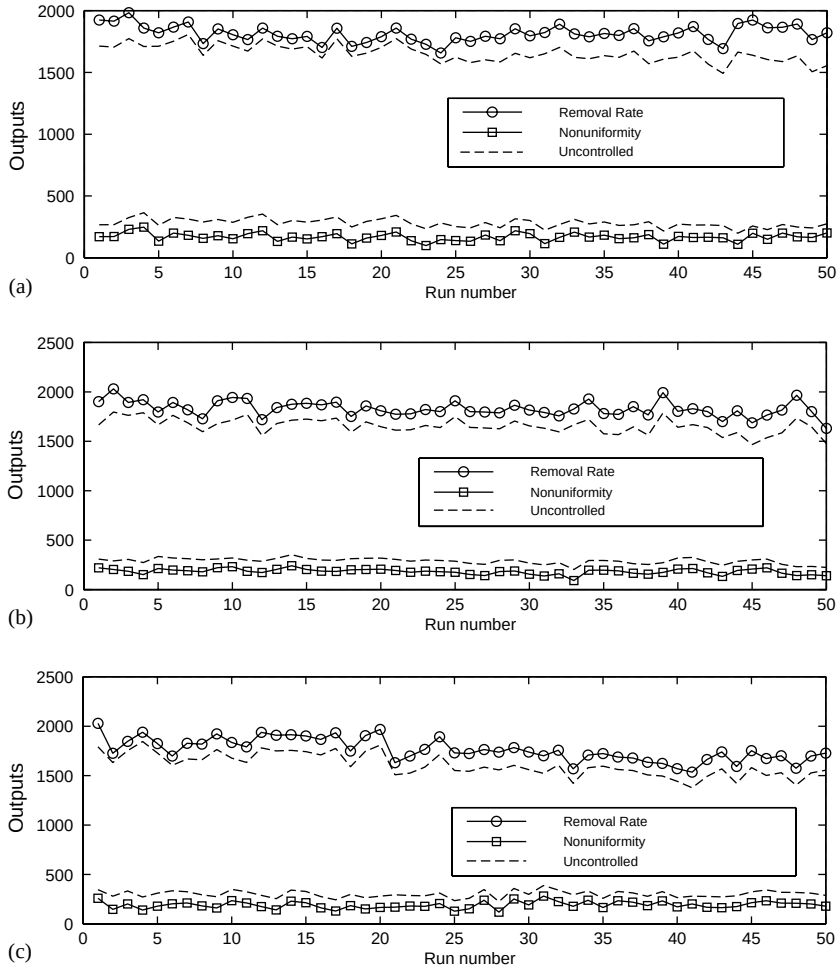


Fig. 7. A CMP process controlled by the DHOBE-MR controller. The outputs are the removal rate and the non-uniformity respectively. (a) Scenario 1: A quadratic model was used to approximate the underlying process. (b) Scenario 2: A linear model was used to approximate the underlying process. (c) Scenario 3: A step disturbance with magnitude −100 happened to response 1 at run 20; another step disturbance with magnitude 50 happened to response 2 at run 30.

The model is rather complex as it includes both quadratic and two factor interaction terms. The inputs u_1, u_2, u_3 and u_4 are scaled to fit in the range $[-1, 1]$. The target values for y_1 and y_2 may be unrealistic (Castillo & Yeh, 1998). These values are set in order to evaluate the performance of the algorithm. For y_1, the larger the value, the better the performance; and for y_2, the smaller the value, the better the performance.

Following (Castillo & Yeh, 1998) to approximate the underlying non-linear process, we used exactly the same two reduced models as in (Castillo & Yeh, 1998), a quadratic form model and a linear form model, respectively. These reduced models provide us with an opportunity to test the controllers' robustness to model errors for non-linear processes.

(2) *Approximate the underlying process—a quadratic model* (*scenario 1*). The 'real' process model of Eqs. (23) and (24) was unknown to the DHOBE-MR controller. As in (Castillo & Yeh, 1998), the following quadratic model was used to approximate the 'real' process:

$$y_1 = 1600 + 150u_1 - 40u_2 + 180u_3 + 25u_4 - 30u_1^2 - 20u_2^2 - 25u_3^2 - 60u_1u_2 - 0.9t \quad (25)$$

and

$$y_2 = 250 + 30u_1 + 100u_2 + 20u_3 + 35u_4 - 30u_1u_2 + 0.05t. \quad (26)$$

As is easily seen, the approximate model is different from the underlying process model, which means that there exists a model error at the beginning of the control experiment. Moreover, the noises in Eqs. (23) and (24) were unknown to the DHOBE-MR controller, so that the controller had to compensate for such disturbances by post-measurements. Our simulation results of the process (simulated by Eqs. (23) and (24)) controlled by the DHOBE-MR controller (designed using the reduced model of Eqs. (25) and (26)) are shown in Fig. 7(a). The two dashed lines in the plot

Table 2
Performance measure (*RMSD*s) of the OAQC and the DHOBE-MR controller for the three scenarios. The response target values for y_1 and y_2 are equal to 2000 and 100, respectively. (The data for the OAQC are from Castillo and Yeh (1998).)

Scenario	Method	$\bar{y}_1$	$\bar{y}_2$	S_{y1}	S_{y2}	$RMSD_1$	$RMSD_2$
1	OAQC	1719.7	168.4	70.4	40.1	288.9	79.2
1	DHOBE-MR	1754.7	157.3	84.5	35.0	259.7	67.5
2	OAQC	1718.2	165.7	72.1	42.0	291.0	78.2
2	DHOBE-MR	1781.9	165.0	84.5	36.1	234.2	74.8
3	OAQC	1661.2	189.2	89.2	43.5	350.2	99.2
3	DHOBE-MR	1741.4	189.1	108.7	35.6	280.8	96.0

are the outputs of the uncontrolled process. The solid lines in the plot with symbols (i.e., circles or squares) depict the controlled outputs. One can see that the controlled process outputs (the removal rate and the non-uniformity) are closer to the targets than the uncontrolled outputs during the entire simulation run.

(3) *Approximate the underlying process—a linear model* (*Scenario 2*). In this scenario, following Castillo and Yeh (1998), we used a linear model to fit the underlying process

$$y_1 = 1600 + 150u_1 - 40u_2 + 180u_3 + 25u_4 - 0.9t, \quad (27)$$

$$y_2 = 250 + 30u_1 + 100u_2 + 30u_3 + 35u_4 + 0.05t. \quad (28)$$

As the underlying process is approximated well by the linear model (based on the results of (Castillo & Yeh, 1998) and our own simulations), the DHOBE-MR controller based on this model also performs well (Fig. 7(b)).

(4) *A quadratic model with step disturbances* (*Scenario 3*). In this simulation, following Castillo and Yeh (1998), two shifts (step disturbances) were fed into the underlying process. The quadratic initial model was used and the constraints were the same as before. The shift for the first response y_1 happened at $t = 20$ with magnitude -100. At t=30, another shift occurred with magnitude 50 for y_2.[7] Our simulation results are shown in Fig. 7(c). The DHOBE-MR controller performs well in this case also.

(5) *Performance analysis*. The final results with regard to the statistical variance analysis are listed in Table 2.[8] The following data are also listed in Table 2 for the convenience of comparison.

- $\bar{y}_i$: the mean of the sampling values from the ith output of the 'real' process.
- S_{y_i}: the standard deviation of the ith output of the 'real' process. The smaller its value, the better.

Table 2 shows that the mean values of the process responses (outputs) controlled by the DHOBE-MR controller are closer to the target values than those of the OAQC. The *RMSD*s of the outputs controlled by the DHOBE-MR controller are smaller than those of the OAQC. Only the standard deviations of response 1 controlled by the DHOBE-MR controller are larger than those of the OAQC. But standard deviation is not the performance metric of interest. Therefore, the DHOBE-MR controller performs slightly better than the OAQC in all scenarios. For more comparisons of the DHOBE-algorithm-based RtR controllers with the OAQC, we refer the reader to (Deng, 1999).

4. Conclusions

In this paper, the DHOBE algorithm is used to estimate process model parameters. Depending on how we apply the DHOBE algorithm leads to different control schemes. If the vector center of the ellipsoid is taken as the estimate of the parameter vector, then it leads to the DHOBE-MR controller. If we instead search for the recipe to minimize the largest expected cost within the feasible set, then the DHOBE-SV controller results. Applying the DHOBE algorithm reduces significantly the computational cost for the set-valued method to estimate the process model. The rescue procedure of the DHOBE algorithm ensures that under large disturbances, the controller will return the process outputs to targets quickly. The DHOBE-MR controller is simpler and less conservative than the DHOBE-SV controller, which tries to minimize the worst-case cost. Simulation results show that the DHOBE-MR controller generally outperforms the DHOBE-SV controller. Therefore, we recommend to use the DHOBE-MR controller in applications.

Compared with the EWMA controller and the OAQC, the DHOBE-MR controller has comparative or even better performance under various conditions in our simulations. The major advantage of the DHOBE-MR controller may be its ability to compensate for large model errors and shifts. Finally, the noise bound is usually unknown in real life. Therefore, we usually need to find the proper noise bound by off-line experiments. There is an adaptive algorithm to find the noise bound on-line for the DHOBE algorithm (Deller, Nayeri, & Odeh, 1993). We will integrate it into the DHOBE-MR controller in the near future.

[7] The magnitudes of these shifts were too small to be discerned with the other noises in (Castillo & Yeh, 1998). However, to make the comparison fair, we used the same values as in (Castillo & Yeh, 1998).

[8] The data on the OAQC performance provided here follow precisely the results in (Castillo & Yeh, 1998).

References

Baras, S. J., & Patel, S. N. (1996). Designing response surface model-based run-by-run controllers: A worst case approach. *IEEE Transactions on Components Packaging and Manufacturing Technology, Part C, Manufacturing*, *19*(2), 98–104.

Boning, D., Moyne, W., & Smith, T. (1995). Run by run control of chemical-mechanical polishing. *1995 IEEE/CPMT International Electronics Manufacturing Technology Symposium* (pp. 81–87).

Butler, S. W., & Stefani, J. A. (1994). Supervisory run-to-run control of polysilicon gate etch using in situ ellipsometry. *IEEE Transactions on Semiconductor Manufacturing*, *7*(2), 193–201.

Castillo, E. D., & Yeh, Y. J. (1998). An adaptive run-to-run optimizing controller for linear and nonlinear semiconductor processes. *IEEE Transactions on Semiconductor Manufacturing*, *11*(2), 285–295.

Cheung, F. M., Yurkovich, S., & Passino, M. K. (1991). An optimal volume ellipsoid algorithm for parameter set estimation. *Proceedings of the 30th conference on decision and control*.

Dasgupta, S., & Huang, F. Y. (1987). Asymptotically convergent modified recursive least-squares with data-dependent updating and forgetting factor for systems with bounded noise. *IEEE Transactions on Information Theory*, *IT-33*(3), 383–392.

Deller, J. R., Nayeri, M., & Odeh, S. F. (1993). Least-square identification with error bounds for real-time signal processing and control. *Proceedings of the IEEE*, *81*(6), 815–849.

Deng, H. (1999). *Run to run controller for semiconductor manufacturing*. Master Thesis, University of Maryland, College Park.

Hamby, E. S., Kabamba, P. T., & Khargonekar, P. (1998). A probabilistic approach to run-to-run control. *IEEE Transactions on Semiconductor Manufacturing*, *11*(4), 654–669.

Hankinson, M., Vincent, T., Irani, K., & Khargonekar, P. (1997). Integrated real-time and run-to-run control of etch depth in reactive etching. *IEEE Transactions on Semiconductor Manufacturing*, *10*, 121–130.

Ingolfsson, A., & Sachs, E. (1993). Stability and sensitivity of an EWMA controller. *Journal of Quality Technology*, *25*, 271–287.

McCarthy, G. S., & Wells, B. R. (1997). Model order reduction for optimal bounding ellipsoid channel models. *IEEE Transactions on Magnetics*, *33*(4), 2552–2568.

Ning, Z. (1996). A comparative analysis of run-to-run control algorithms in the semiconductor manufacturing industry. *1996 IEEE/SEMI Advanced Semiconductor Manufacturing Conference* (pp. 375–381).

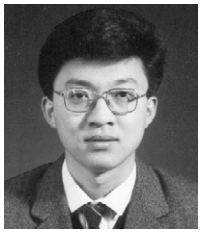
Palmer, E., Ren, W., & Spanos, C. J. (1996). Control of photoresist properties: A Kalman filter based approach. *IEEE Transactions on Semiconductor Manufacturing*, *9*(2), 208–214.

Rao, K. A., & Huang, Y. (1993). Tracking characteristics of an OBE parameter-estimation algorithm. *IEEE Transactions on Signal Processing*, *41*(3), 1140–1148.

Sachs, E., Hu, A., & Ingolfsson, A. (1995). Run by run process control: Combining SPC and feedback control. *IEEE Transactions on Semiconduct. Manufacturing*, *8*, 26–43.

Smith, T. H., & Boning, D. S. (1997). A self-tuning EWMA controller utilizing artificial neural network function approximation techniques. *IEEE Transactions on Semiconductor Manufacturing*, *20*(2), 121–132.

Chang Zhang received the B.Eng. degree from Dept. of Automation and the BS degree from Dept. of Business Management in Tsinghua University, Beijing, China, in 1994. In 1997, he received the M.Eng. degree from Dept. of Automation, Tsinghua University. He obtained MS degree from Electrical and Computer Engineering department, University of Maryland, College Park, USA in 1999. Currently, he is a Ph.D. candidate and research assistant at University of Maryland, College Park, USA. His research interests include optimization of large-scale communication and control systems, process control and robust control, reinforcement learning, and signal processing.



Hao Deng obtained his BS degree from the Automation Department of Tianjin University, Tianjin, PR China in 1991. Afterwards he worked for Honeywell China Inc as an Engineer. From 1997, he began studying towards the MS degree in Electrical and Computer Engineering Department of University of Maryland at College Park, Maryland, USA. At the same time, he worked as a research assistant for the "Run by Run Control for Semiconductor Manufacturing" project. He received his MS degree in 1999 from University of Maryland at College Park with a thesis on the same subject. Currently, he is a control system engineer in Bechtel Power Corporation, USA.

John S. Baras was born in Piraeus, Greece, on March 13, 1948. He received the B.S. in Electrical Engineering with highest distinction from the National Technical University of Athens, Greece, in 1970. He received the M.S. and Ph.D. degrees in Applied Mathematics from Harvard University, Cambridge, MA, in 1971 and 1973, respectively. Since 1973 he has been with the Electrical and Computer Engineering Department, and the Applied Mathematics Faculty, at the University of Maryland, College Park, where he is currently a Professor holding a permanent joint appointment with the ISR. He is also an Affiliate Professor of Computer Science. From 1985 to 1991 he was the Founding Director of the Institute for Systems Research. On February 1990 he was appointed to the Lockheed Martin Chair in Systems Engineering. Since 1991 he has been the Director of the Center for Satellite and Hybrid Communication Networks, a NASA Center for the Commercial Development of Space, which he co-founded. Dr. Baras has held visiting research scholar positions with Stanford, MIT, Harvard University, the Institute National de Reserche en Informatique et en Automatique, and the University of California Berkeley.

He has numerous publications in control and communication systems, and is the co-editor of Recent Progress in Stochastic Calculus, Springer, 1990. Dr. Baras holds two patents and has four patent applications pending. Professor Baras' research interests include satellite and hybrid communication networks, integrated network management systems, fast Internet services via hybrid, satellite and wireless networks, network security, stochastic systems, robust control of nonlinear systems, real-time parallel architectures for nonlinear signal processing, intelligent control systems, expert and symbolic systems for control and communication systems synthesis, distributed parameter systems, planning and optimization, real-time architectures for intelligent control, speech and image understanding, biomimetic algorithms and systems for signal processing and sensor networks, intelligent manufacturing of smart materials, integrated product-process design.

Among his awards are: 1978, 1983 and 1993 Naval Research Laboratory Research (Alan Berman) Publication Awards; the 1980 Outstanding Paper Award of the IEEE Control Systems Society; 1991 and 1994 Outstanding Invention of the Year Awards from the University of Maryland; Outstanding Paper Award, "ATM in Hybrid Networks", at Design SuperCon 1996; 1996, MIPS Research Award of Excellence for Outstanding Contributions in Advancing Maryland Industry; 1998, the Mancur Olson Research Achievement Award, from the University of Maryland College Park.

Professor Baras is a Fellow of the IEEE. He has consulted extensively with industry and government on various automation and telecommunication problems. He has served in: Board of Governors of the IEEE Control Systems Society; IEEE Engineering R& D Committee; Aerospace Industries Association advisory committee on advanced sensors; IEEE Fellow evaluation committee. He is currently serving on the editorial boards of Mathematics of Control, Signals, and Systems, of Systems and Control: Foundations and Applications, of IMA J. of Mathematical Control and Information, of Systems Automation-Research and Applications.