ABSTRACT

Title of Thesis: RUN BY RUN CONTROL FOR SEMICONDUCTOR MANUFACTURING

Name of degree candidate: Hao Deng

Degree and year: Master of Science, 1999

Thesis directed by: Professor John S. Baras
Department of Electrical Engineering

A new type of Run-by-Run controller based on the DHOBE (Dasgupta-Huang Optimal Bounding Ellipsoid) algorithm is designed and simulated for semiconductor manufacturing process. One approach is to use the algorithm to implement online model identification which leads to a model-reference controller. The other approach is utilizes the worst case idea, to implement the set-valued controller. Both kinds of controllers are applied to linear and quadratic models which are derived from experiments. The controllers are simulated for cases when processes are satisfy slow drifting, abrupt shift, bad data and model errors. The controllers are tuned according to the requirements of the algorithm and process and the simulation data is analyzed according to the performance benchmark. All the simulation results are compared to either the Exponentially Weighted Moving Average (EWMA)or Optimal Adaptive Quality Controller (OAQC) control method.

RUN BY RUN CONTROL FOR SEMICONDUCTOR MANUFACTURING

by

Hao Deng

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland at College Park in partial fulfillment
of the requirements for the degree of
Master of Science
1999

DEDICATION

To my parents

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to my advisor Dr. John S. Baras for his encouragement, guidance and support during the past two years of my graduate research. It is both a privilege and pleasure to be his student.

I would like to thank Dr. William S. Levine and Dr. Evanghelos Zafiriou for serving on my thesis advisory committee.

I would also like to thank my friend and fellow student Chang Zhang for many helpful discussions on the problems addressed in this thesis. Thanks to Tina Vigil for her continuous support.

I am especially grateful to my wife, Yi Sun for her love and support during the past two years.

Acknowledgment of support by SRC Contract: 97-F5-466 and by NSF Grant: ECS-97-27805.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Semiconductor Manufacturing Processes and Models

1.1 Semiconductor Manufacturing Processes

Semiconductor devices have entered the VLSI age and are being routinely used in a wide range of applications and in fields quite different from traditional electronics. This has meant that production rate and quality has had to increase continuously to match consumption demand. This increase in production volume and application variety has raised the need for dramatic improvements in integration density, performance, reliability and cost reduction. The realization of these improvements requires better control of the process and of the equipment. For instance, the increasing complexity of semiconductor devices enables the packing of individual devices closer together on a single chip and at the same time requires improvements in the precision and quality of the control techniques of the processes. This, among other things, means reducing the process variation in all phases of production. On the other hand, the increasing complexity translates also to longer processing times. In practice, it normally takes 30-40 days and nearly 100 steps to finish the manufacturing of 1 lot of wafers[13].

Other factors also affect the manufacturing process:

- Due to improvements in equipment, there tends to be frequent replacement of equipment which in turn leads to frequent tuning of the process' operating condition.
- There is generally an insufficient number of on-line sensors and actuators at each process step to establish a satisfactory control over process parameters.
- Lack of accurate and detailed knowledge of the process behavior often leads normal (conventional) control schemes to difficulties.

Although there are various kinds of semiconductor manufacturing processes, they are mainly based on the following elementary technologies: wafer cleaning and preparation, oxidation, ion implantation, thin film formation, lithography and etching. The formation of the circuits' functions on the wafer is by successive application of these steps. Failure in any unit process will result in the failure of the whole lot which means low yield.

With all the above difficulties, the required investment in a typical semiconductor manufacturing line is normally very high. And it would be natural to reach the point that the yield of the production is becoming extremely important. A promising feasible solution to the attainment of high yield appears to be robust control. The most popular process control method in the semiconductor manufacturing industry is the Run-by-Run(RbR), control which will be covered at length in the following sections. The popularity of RbR control in semiconductor manufactoring is due to two reasons primarily: (a) it is the closest method to real-time control which can be feasibly implemented and accepted by the industry; (b) it is a good fit to the "batch" manufacturing typically employed in the semiconductor factories. The Dasgupta-Huang Optimal Bounded Ellipsoid (DHOBE) controller based on model-reference and set-valued methods will be discussed and its performance will be evaluated using simulations. The performance of this method was analyzed and compared with other RbR control methods.

1.2 Semiconductor Manufacturing Process Models

1.2.1 Introduction

During each semiconductor processing step, a wafer is contained within some physical environment, that has been generated by a piece of fabrication equipment within a facility as a result of settings, which are controlled or dictated by a program or recipe.

The first step in developing a model-based control system consists of developing a regression model that relates the controllable variables with the quality characteristic of interest. This model often takes the form of recipes. A state description is a set of state variables and the model is the knowledge of how a state set is affected by itself and other states and inputs. A model can be

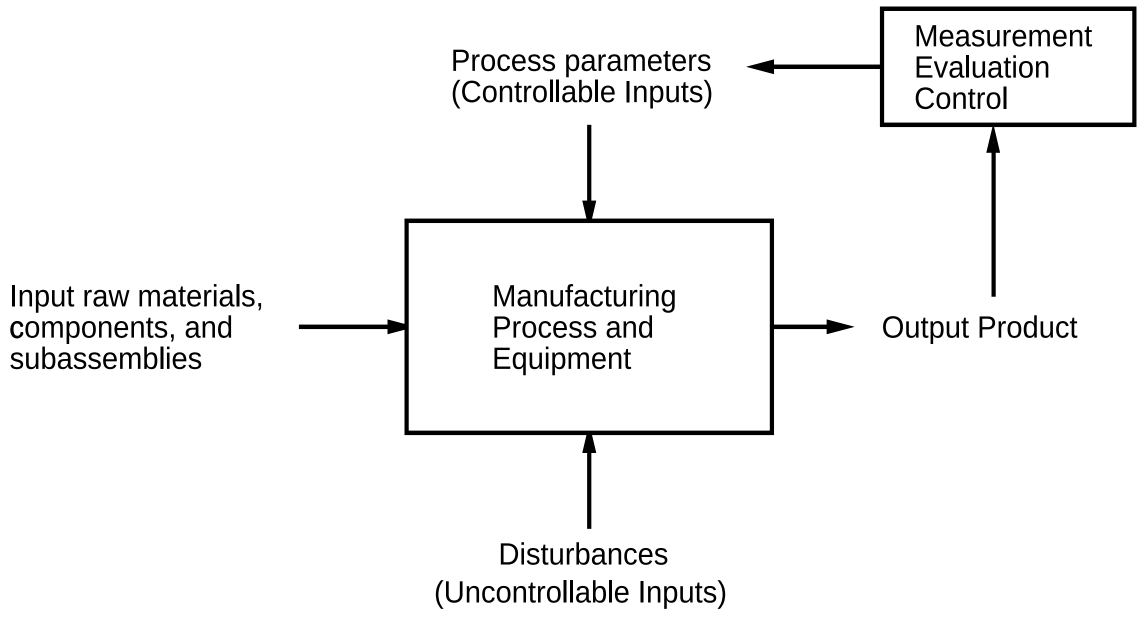

Figure 1.1: Semiconductor Manufacturing Process Model and Control[14]

described as a function or a map taking some number of input states to some number of output states[14].

In most control applications, some prior information from designed experiments is usually available. A model relating the controllable variable and the responses can provide the initial estimate of the models' parameters. From these experiments, an initial recipe can be determined. The better the initial estimates of model parameters and initial recipes are, the smaller the transient effect at startup. Usually due to the lack of knowledge of the process, empirical models have to be used for control purposes. Sometimes the initial models are estimated off-line after designed experiments.

A basic assumption in this step is that the process exhibits no dynamics.

This means that the quality characteristic (response) at run t, depends only on the recipe or input variables at the start of run t. No previous inputs or previous outputs will have an effect on the current output. The second assumption is that there is no delay between the control action and its effect on the response. The final result of modeling is a set of linear regression models that relates inputs to outputs and a recipe that achieves a desirable initial performance. Once the process is optimized, the second step is applied to maintain the process as close as possible to optimum, which becomes the target value of the quality characteristic.

1.2.2 Disturbances of the Processes

The processes may be subject to different shifts (in the values of their parameters), noises, and other disturbances.

Normally the noises are modeled in simulations as white noise with specific variance. The most commonly encountered situations in the production settings are:

1. The process is drifting slowly, say on the order of 1σ over a period of 100-1000 runs.
2. The process is subject to occasional large shifts, say on the order of 2σ in the runs, which occur after maintenance operations or specification changes.
3. Bad data from the sensors will cause the controllers to take fault control action. This can also be considered as a disturbance to the processes.
4. In simulations, it is found that the model error can also be considered as a perturbation to the processes.

1.2.3 Model Structure

While the controller has requirements for the predictive capability of the model, it has no obvious constraints on the form of the model. The easily obtainable model structure suitable for control is a low-order polynomial model which is usually derived using statistically designed experiments RSM(Response Surface Methodology).
Non-linearities can be encountered either at the inputs or at the outputs of the system. When the non-linearities happen at the output, in general, it seems reasonable to use a low-order polynomial to approximate the nonlinear effects of the transfer function. Alternatively, non-linearities can be encountered at the inputs, a typical case being what is called a Hammerstein model. The latter case is easier to handle since such models are linear in the parameters and therefore, recursive least square estimation can be applied to obtain on-line estimates[5].

Developers of linear control algorithms believe [12] that most processes can be described using linear models. This view is based on the belief that although the control model is not restricted to a first order model, since the task of the controller is to maintain the process near an operating point after it has been optimized, the control actions will tend to be restricted to a small range of operating space. Therefore, it is commonly accepted in practice that first order models will be sufficient to summarize the local process behavior in most cases. However, there are many semiconductor manufacturing processes where non-linearities are an important feature. If non-linearities are severe, a linear controller might maintain stability only for a few runs. In general, if non-linearities can be modeled and taken into account in the design of a controller, better performance will be achieved over the one obtained with linear models.

In the following chapters, we did not differentiate the control methods and the forms of the models. As the DHOBE can handle either linear or nonlinear models, Hammerstein model form is used in general to test the algorithms.

1.2.4 Chemical Mechanical Planarization(CMP)

With the complexity of Very Large Scale Integrated circuits increasing, their Critical Dimension(CD) decreases and the number of levels in the devices increases. Planarization techniques become more and more important to the success of devices using a multi-level interconnect architecture, and for satisfying increasingly stringent performance requirements. This is because for smaller CDs and larger device sizes, higher resolution is needed for patterning circuits, resulting in a smaller depth of focus of optical steppers, thus requiring a globally planarized surface. Moreover, for multi-level devices, Planarization is critical to prevent the surface from becoming more nonplanar with each additional level. CMP, as a newly developed planarization technique which can meet the planarization requirements of decreasing CDs and multi-level devices, is demonstrated to be the only global planarization technique available and is considered to be a strategically important technology for next generation multi-level devices. The application of the CMP process range from the planarization of oxide films to defect reduction[16].

The wafer is held face down by a carrier which presses the wafer against a polishing pad mounted on a rotating rigid platen. A slurry, consisting of abrasive

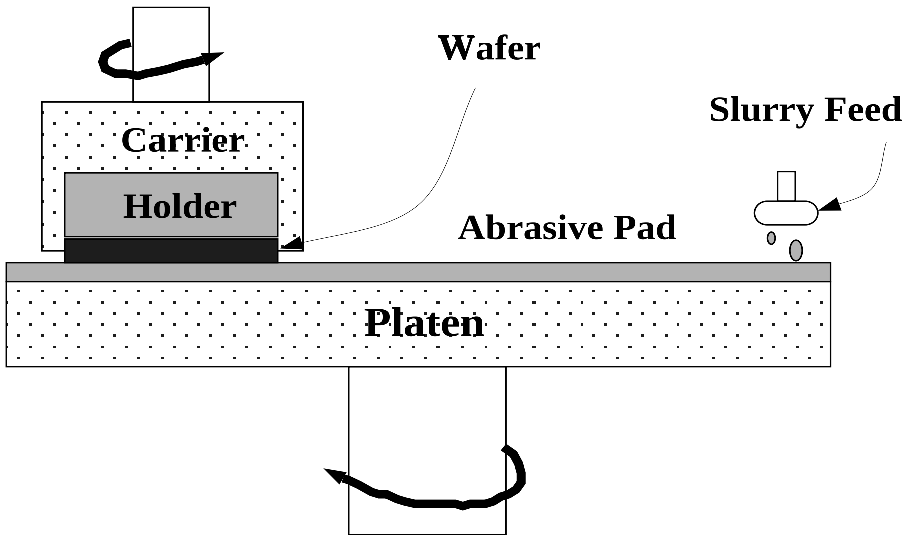

Figure 1.2: Illustration of CMP Processing [15]

colloidal particles kept in suspension in water, is delivered to the surface of the polishing pad through a slurry tube. Planarization is achieved when mechanical abrasion removes material from the wafer surface in such a way that high areas on the wafer undergo greater removal rate than low areas [15].

The CMP process offers significant advantages over conventional planarization techniques because it is a global planarization technique and because it is the only fabrication technique which can actually reduce defect density. However the control of the CMP is chronically poor, arising from the poor understanding of the process, degradation of polishing pads, inconsistency of the slurry, and the lack of in-situ sensors. Because the process includes the mechanical abrasion of the surface, the polishing pad wears rapidly; hence the removal rate continuously decreases. Furthermore, there is no reliable real time sensor currently available for the CMP process; process control has to be based on post-process measurements. All of these features provide an opportunity for effective run-by-run control schemes and our simulation experiments are concentrated on this process.

There are many process models summarized for algorithm development and comparative analysis. In [6] two kinds of models, linear and second order, of CMP are given as follows. The four controllable inputs to the model are Platen Speed (u_1), Back Pressure (u_2), Polish Head Down-force (u_3) and Profile (u_4). The two outputs are Removal Rate (y_1) and Non-uniformity (y_2).

(1) Linear Model

The linear model has the form of

$$y[n] = C + Au[n] + \omega[n] + \delta[n] \tag{1.1}$$

where

$$C = \begin{bmatrix} -1382.60 \\ -627.32 \end{bmatrix}$$

$$A = \begin{bmatrix} 50.18 & -6.65 & 163.4 & 8.45 \\ 13.67 & 19.95 & 27.52 & 5.25 \end{bmatrix}$$

$$\delta = \begin{bmatrix} -17 \\ 1.5 \end{bmatrix}$$

$$\Lambda = \begin{bmatrix} 665.64 & 0 \\ 0 & 5.29 \end{bmatrix},$$

$\omega[n]$ is normally distributed white noise with zero mean and covariance matrix Λ. u is the vector of controllable inputs.

(2)Second Order Model

The second order model has the form of

$$y[n] = C + f(u[n]) + \omega[n] + \delta[n] \tag{1.2}$$

where

$$f(u[n]) = \sum_{i=0}^{3}\sum_{j=0}^{3} \beta(i,j)u(i)u(j)$$

$$\begin{aligned}\beta \;=\; & \left[\begin{array}{cccccccc} 1386.5 & 381.02 & 112.19 & 3778.8 & -21.301 & 8.7159 & 24.953 & 37.082 \\ 1520.8 & 2365.6 & 2923.5 & 281.66 & -3.9419 & -1.0754 & 1.406 & 0.33797 \end{array}\right. \\ & \left.\begin{array}{cccccc} -17.642 & -14.974 & -164.99 & 28.150 & 249.17 & 0.025067 \\ -72.274 & -94.222 & -26.175 & 13.505 & 36.691 & 32.929 \end{array}\right]\end{aligned} \tag{1.3}$$

There is another series of models from experimental data summarized in [5].

(1) Almost Linear 4 $\times$ 2 CMP Model

The first one is called almost linear model as its form is in second order but all the coefficients of the second order terms are comparatively small.

$$\begin{aligned} y_1 \;=\; & 1563.5 + 159.3u_1 - 38.2u_2 + 178.9u_3 + 24.9u_4 - 67.2u_1u_2 - 46.2u_1^2 \\ & -19.2u_2^2 - 28.9u_3^2 - 12u_1t' + 116u_4t' - 50.4t' + 20.4t'^2 + \epsilon_{1,t} \end{aligned} \tag{1.4}$$

$$\begin{aligned} y_2 \;=\; & 254 + 32.6u_1 + 113.2u_2 + 32.6u_3 + 37.1u_4 - 36.8u_1u_2 + 57.3u_4t' \\ & -2.42t' + \epsilon_{2,t} \end{aligned} \tag{1.5}$$

where

$t' = (t-53)/53$, $\epsilon_{1,t} \sim N(0, 60^2)$, $\epsilon_{2,t} \sim N(0, 30^2)$

All the input variables are normalized to $[-1, 1]$.

(2) Nonlinear 3 $\times$ 2 CMP Model

This model uses only 3 input factors: Back Pressure Down-force (u_1), Platen Speed (u_2) and Slurry Concentration (u_3). The two outputs of the model are Removal Rate (y_1) and Within-wafer Standard Deviation (y_2). It is fully quadratic model with sever nonlinearity.

$$\begin{aligned} y_1 &= 276.5 + 574.6u_1 + 616.3u_2 - 126.7u_3 - 1109.5u_1^2 - 286.1u_2^2 + 989.1u_3^2 \\ &\quad -52.9u_1u_2 - 156.9u_1u_3 - 550.3u_2u_3 - 10t + \epsilon_{1,t} \end{aligned} \tag{1.6}$$

$$\begin{aligned} y_2 &= 746.3 + 62.3u_1 + 128.6u_2 - 152.1u_3 - 289.7u_1^2 - 32.1u_2^2 + 237.7u_3^2 \\ &\quad -28.9u_1u_2 - 122.1u_1u_3 - 140.6u_2u_3 + 1.5t + \epsilon_{2,t} \end{aligned} \tag{1.7}$$

1.2.5 Photoresist Spin-Coat and Bake Operation of Lithography

Lithography is a process whereby the circuit pattern is mapped on the surface of a silicon wafer in the form of a mask. The first step is to cover the whole wafer surface with a thin layer of resin-photo resist. The coat should be uniform and free of defects such as pinholes. This is not easy, especially when the surface is uneven because of the patterns formed in earlier stages. The spin-coat method is normally used under the following procedures.

The wafers are held on a vacuum chuck and the resists are mixed with solvents to adjust the fluidity and then sprayed on to the surface. The chuck is then rotated at a certain speed to remove excess material and form a uniform coat. Then the resist layers are dried by heating in an oven (soft baking) at a certain temperature for a certain period of time. The wafers then undergo exposure, developing, and hard-baking phases to form the pattern and remove all the un-useful solvent in the resin[13].

The following model was summarized in [11]. The inputs and their ranges are Spin Speed (SPS) 4500-4700 RPM, the Spin Time (SPT) 15-90 sec, the Baking Temperature (BTE) 105-135 Degree C and the Baking Time (BTI) 20-100 sec. The responses of the model are Resist Thickness (T) and Reflectance (R). And the model from the experimental data is:

$$T = -13814 + \frac{2.54 \cdot 10^6}{\sqrt{SPS}} + \frac{1.95 \cdot 10^7}{BTE\sqrt{SPS}} - 3.78BTI - 0.28SPT - \frac{6.16 \cdot 10^7}{SPS} \quad (1.8)$$

and

$$\begin{aligned} R &= 134.4 - 0.046SPS + 0.32SPT - 0.17BTE + 0.023BTI - 4.34 \cdot 10^{-5} \\ &\cdot SPS \cdot SPT + 5.19 \cdot 10^{-5} \cdot SPS \cdot BTE - 1.07 \cdot 10^{-3} \cdot SPT \cdot BTE \\ &+ 5.15 \cdot 10^{-6} \cdot (SPS)^2 - 4.11 \cdot 10^{-4} \cdot SPT \cdot BTI \qquad (1.9) \end{aligned}$$

Although there are many processes in the semiconductor manufacturing industry, as long as they can be expressed as linear in the parameter models, they are in the same family of problems for the control scheme. Thus what is of interest to us is the structure of the process model rather than the specific kind of process.

The DHOBE (Dasgupta-Huang Optimal Bounded Ellipsoid) RbR controller invented in this thesis is capable of handling linear and nonliner models. The controller's compensation for drifts, large step disturbances and large model errors are significant and the resulting performance has been compared with other popular control methods such as EWMA (Exponentially Weighted Moving Average) and OAQC (Optimal Adaptive Quality Controller).

Chapter 2

Run-by-Run Controllers

2.1 General Introduction of RbR Controllers

Many manufacturing systems are controlled with PID type controllers. In industries like semiconductor manufacturing, specifications or changing conditions impose a need for adjusting such controllers on a Run-by-Run (RbR) basis. This need has originated a collection of techniques called Run-by-Run process control.

RbR control is a form of discrete process and equipment control in which the product recipes with respect to a particular equipment process are modified ex-situ, i.e. between equipment "runs", so as to minimize the effects of process drift, shift, and other variabilities to keep the outputs at prescribed target values.

The most widely used RbR controllers are model-reference controllers such as the EWMA, OAQC and IMC. The description of model-reference RbR controllers will be covered in sections 1 and 2 of this chapter. The Set-Valued method of RbR control will be introduced in section 3 of this chapter.

The model-reference RbR controller is designed in the following way: First, it performs process control based only on post-process measurements. Then it responds to post-process measurements by updating models of the process between runs. Finally it provides a new recipe for use in the next run of the process. It does not modify the recipe during a run based on measurements made while the process is running. The RbR controller's capability to update the recipe for each run is mainly based on the fact that many control parameters in semiconductor fabrication can be changed between runs with little or no cost or time delay. The reason why it generates new recipes from the post-process measurements on a run-by-run basis is, on one hand, lack of online sensors for the process. On the other hand, frequent changes of inputs to the process will only increase the variability of the process output. Sometimes deadband has to be utilized in order to make the change less frequent.

The model-reference RbR controller usually has two parts: Optimization of the model and recipe generation. For all the control strategies, the key issues are modeling, stability, and performance analysis.

The process control models need to account for the process variations throughout the whole process, providing enough accuracy for control without too many adjustable parameters. Typically the process model for RbR controller is static and empirical, where the model structure is polynomial in the inputs. The initial models are derived from former off-line experiments. These are the best estimates of the model before the controller is put into operation. After the

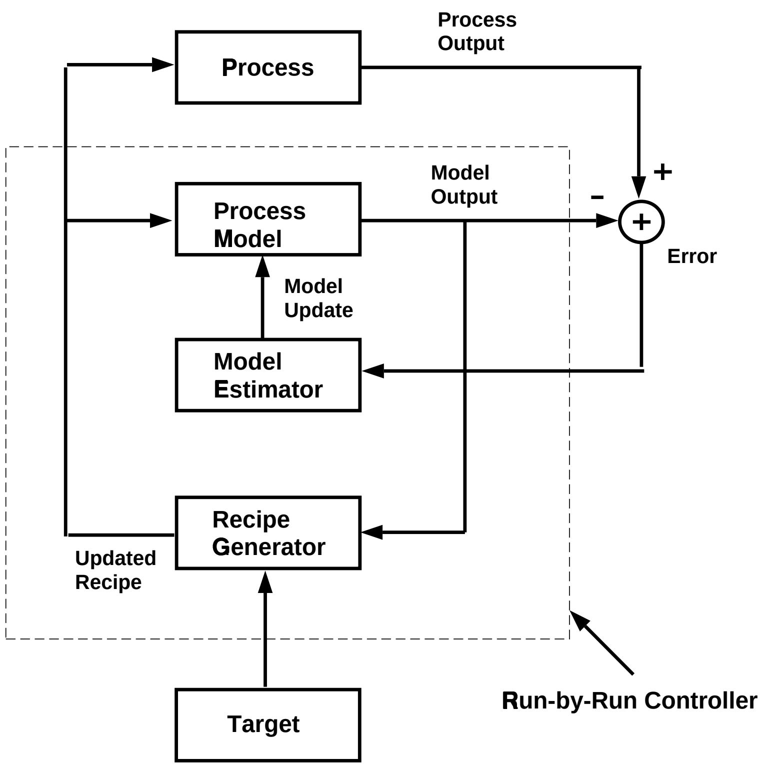

Figure 2.1: Diagram of RbR Controller

controller is online, the model within the controller is updated using the new measurements of the output of the process run-by-run. Various types of model updating methods lead to different kinds of controllers which will be introduced in the following sections.

After the model is updated, the next step is to use it to predict the output of the process and generate the recipe for the next run in order to minimize the variation of the process response. A typical block diagram of an RbR controller is illustrated in Figure 2.1. Normally the goal of the RbR controller is to reduce the variability of the process output as measured by the mean squared deviation from the target. The control feedback from an RbR controller is sometimes accomplished by suggesting target output values to the next lower level of control, i.e. the RbR controller can be used as a supervisory controller which provides the setpoints to the real-time controllers. In the absence of lower level control, the RbR feedback would consist of suggestions for equipment input parameter settings.

2.2 EWMA Gradual Mode RbR Controller

2.2.1 Introduction

The Exponentially Weighted Moving Average (EWMA) RbR controller was first developed by E. Sachs et al. from MIT [12]. Its gradual mode mainly compensates for slowly drifting processes that can be represented by linear models.

The EWMA is a statistic with the characteristic that it gives less and less weight to data as they get older and older . It has the desirable feature of allowing the weight of a point to decay gradually with age in a geometric fashion. It can be used as a dynamic process control tool. To control a process, it is convenient to forecast where the process will be in the next instance of time. Then if the forecast shows a future deviation from target that is too large, some remedial control action from the controller or the process operator will compel the forecast to equal the target. In semiconductor manufacturing industry, the observation is recorded on every piece manufactured, a forecast based on the unfolding historical record can be used to initiate a feedback control loop to adjust the process.

The EWMA gradual mode controller is a linear approximation model-based controller with an EWMA forgetting factor that provides for control of noisy processes. The controller only updates the offset term of the model on a Run-by-Run basis using EWMA. By adjusting the intercept term, the controller is able to track and compensate for gradual changes in the process as well as filter out random walk noise in the process.

The update of the intercept term constitutes a smoothing (or filtering) of measurements of the intercept term in order to adapt the process model to account for recent changes in the process. The amount of smoothing performed for the output is a function of the EWMA weight . Higher weights indicate recent measurements are weighted more in each update, and therefore indicate less filtering. This method is effective for many processes in the semiconductor industry. This is because many processes are subject to small shift or drift offset changes in the overall equipment state, but the underlying process dependencies do not change.

The linear EWMA controller has been shown to improve run by run process control for approximately linear processes which are subject to shifts or persistent drifts in the presence of noises. But the EWMA controller is unable to adequately control processes which are poorly represented by such models.

2.2.2 The EWMA Algorithm

Suppose that the manufacturing process can be described by a simplified SISO linear model; i.e. the output of the process y_t is linearly related to the process input u_t in the form of $y_t = \alpha_t + \beta u_t$. However, as the real process model is unknown to us, we have the estimated model as $y_t = a_t + b u_t$. Here it is assumed that the slope term of the real and estimated model do not change from run to run. And the model update is only limited to updating the value of the intercept term. Then according to the EWMA algorithm introduced in [12], we have the following recursive relations:

$$u_t = \frac{(T - a_{t-1})}{b}$$

and

$$a_t = \sum_{i=1}^{t} w(1-w)^{t-i}(y_i - b u_i) = w(y_t - b u_t) + (1-w)a_{t-1}.$$

The first equation specifies the generation of the recipes based on model estimation using previous run measurements; where T is the target value for the process output. The second equation specifies the model updating using the measurement of the process output y_t of current run and the previous model parameter estimations. $w \in (0,1)$ is the weight that is assigned to current measurements. When the input to the process is not a scalar but a vector, then the recipe generation equation becomes

$$u_t = \frac{T - a_{t-1}}{b^T b} b + (I - \frac{b b^T}{b^T b}) u_{t-1}$$

This choice of u_t minimizes the change in recipe between runs measured by: $||u_t - u_{t-1}|| = \sqrt{(u_t - u_{t-1})^T (u_t - u_{t-1})}$ The key point of tuning the EWMA RbR controller is to select the appropriate weight w. The necessary and sufficient condition for y_t to converge to T is that

$$0 < \frac{w\beta}{b} < 2,$$

which is to say that the weight should depend on the error of the model from the real process [17]. However, if the weight is selected large, the noise of the process will deteriorate and if weight is small, the compensation result will not be satisfactory. A fixed weight has been used in previous EWMA controllers. Recently, artificial neural networks (ANN) have been incorperated in the EWMA controller in order to change and select the weight on-line according to previous experiences [10].

2.3 Optimizing Adaptive Quality Controller (OAQC)

The OAQC controller is another form of model-reference RbR controller [6], [5] [19]. It can act both as an optimizer and a controller. In its optimizer mode, it updates the model at every run and in the controller mode, it uses a quadratic cost function to maintain the response of the process at the desired target with regards to the variation of the tunable parameters. It integrates the multivariate control chart as a deadband to the controller in order to erase outliers, which are harmful to the control and optimizing actions. This method can be applied to nonlinear models as well. Simulations have been done for second order Hammerstein models of CMP.

The OAQC controller's internal model is summarized as the second-order MIMO Hammerstein transfer function model by the following form

$$y_t = (I_p - L\mathcal{B})y_t = y(0) + Nz_{t-1} + M(t) + (I_p - C\mathcal{B})\epsilon_t$$

where

$$z_t^T = (u_t, u_t^2, u_t^{(i)}, u_t^{(j)}) \qquad (i < j)$$

is a $2n + (n(n-1)/2)$ vector that contains quadratic expansions of u_t.

y_t is a $p \times 1$ vector of model responses.

u_{t-1} is the vector of controllable parameters.

$\mathcal{B}$ is the shifting operator.

$M(t)$ is the shifting term and ϵ is the white noise random vector.

After simplification, the model can be changed to the following form

$$\hat{y}_{t+1|t} = Ly_t + M(t+1) + Nz_t.$$

The model estimation task is to get the on-line parameter estimates of $\hat{L}$, $\hat{M}$,$\hat{N}$ and this is implemented using the multivariate recursive least-squares algorithm.

By redefining the model's vector of regressors and coefficients as the following, the above model is changed to the normalized form

$$\hat{y}_{t+1|t}^{[i]} = \hat{\theta}^{[i]^T} \phi_t^{[i]},$$

where

$$\phi_t^{[i]} = (y_{t-1} | z_{t-1}^{[i]} | t)$$

$$\hat{\theta}^T = [\hat{L} | \hat{N} | \hat{M}].$$

Then the estimation of the model parameter is changed to estimating the vector $\hat{\theta}^{[i]}$ according to the following procedure. Here i is the index of the response.

$$K_t^{[i]} = P_{t-1}^{[i]} \phi_t^{[i]} / (\lambda + \phi_t^{[i]^T} P_{t-1}^{[i]} \phi_t^{[i]})$$

$$e_t^{[i]} = y_t^{[i]} - \phi[i]^T{}_t \hat{\theta}_{t-1}^{[i]}$$

$$\hat{\theta}_t^{[i]} = \hat{\theta}_{t-1}^{[i]} + K_t^{[i]} e_t^{[i]}$$

$$P_t^{[i]} = [I - K_t^{[i]} \phi_t^{[i]^T} P_{t-1}^{[i]}]/\lambda + R_t^{[i]}$$

where

$$R_t^{[i]} = I K_t^{[i]^T} P_{t-1}^{[i]} \phi_t^{[i]} / [2(p+n) + (n(n-1)/2)]$$

After the model is updated, the algorithm will use the prediction of the model output to optimize the cost function of the following quadratic form

$$J = (\hat{y}_{t+1|t} - T)^T W (\hat{y}_{t+1|t} - T) + (u_t - u_{t-1})^T \Gamma (u_t - u_{t-1})$$

where T is the target, W and Γ are diagonal matrices.

2.4 The Set-Valued RbR Controller

The Set-Valued RbR control method was developed by J.S.Baras and N.S.Patel [7]. It is also a model-based method. However, the set-valued controller considers the uncertainty of the model identification. This uncertainty exists because normally the updated model can not be exactly accurate due to model errors, measurement noises and other perturbations. What can be identified more precisely is the set of the models in which the real process model resides. We could be quite certain that the model is somewhere in this set, but due to the randomness of the process, the exact position is unknown. Then the set-valued method together with the worst case approach will select such a model parameter point from the set which makes the cost function the largest (worst case). Then by using these model parameters, the recipe that will optimize the cost function will be found. This recipe is not the best and is somewhat conservative but it is ensured that it fits even for the worst case. Thus it can be applied safely to any model in the model set. The conceptual framework is as follows.

The system model is considered as consisting of several mappings. The state (model parameters) of the current run is mapped from the state of previous run by

$$M_{t+1} \in \mathcal{F}(M_t).$$

The response of the process model is a mapping from the state and recipe by

$$y_{t+1} \in \mathcal{G}(M_t, u_t).$$

The cost function is defined by

$$z_{t+1} = l(y_{t+1})$$

This is a rather general definition of the system as the structure of the mappings and the cost function are left unspecified. The recipe is always generated at the end of each run for the use of next run and the measurement is always for the current run. Then at run j, a feasible set P_j of the system states can be defined. It can be calculated recursively by the following

$$\mathcal{P}_j = \{M \in P_{j-1} : y_j \in \mathcal{G}(M, u_{j-1})\}$$

and

$$P_j = \bigcup_{M \in \mathcal{P}_j} \mathcal{F}(M).$$

The worst case approach of the set-valued method is implemented here to generate the recipe by choosing the state that makes the cost function the largest, and then computes the recipe.

$$\min_{u \in U} \max_{M \in P_j} \max_{y \in \mathcal{G}(M,u)} l(y).$$

The set-valued method has been applied using the Optimal Volume Ellipsoid (OVE) algorithm [20] to get the set of models when the process model is limited to a polynomial structure.

Chapter 3

RbR Controller Based on the DHOBE Algorithm

3.1 The DHOBE Algorithm

3.1.1 Introduction

DHOBE which is known as the Dasgupta-Huang Optimal Bounded Ellipsoid was first developed in 1987 by Dasgupta and Huang [1]. It modified the OBE algorithm of Fogel and Huang [8] by introducing a forgetting factor. It was further developed by Rao and Huang in 1993 [2] by introducing the rescue procedure. This algorithm belongs to a class of bounded-error estimation algorithms termed set-membership parameter estimation algorithms.

It is used in the RbR controller to achieve the online recursive model parameter estimation. When applied to the Set-Valued method, it is used to get the ellipsoid of the model parameters. The largest benefit of this algorithm is that unlike other recursive algorithms, DHOBE discerns if the new measurement contains any fresh information. This will save a lot of calculations and reduces significantly the calculations for load estimation. The only knowledge required from the real process is the strict bound of the noise instead of the distribution of the noise. Another improvement is the introduction of the rescue procedure. As stated in Chapter 1 the semiconductor manufacturing process usually undergoes abrupt shifts (due to equipment maintenance) and modelling errors (considered as big shift at the startup). This normally causes other model identification methods to return an empty set for the parameters. The rescue procedure greatly improves the performance of the algorithm under this circumstance and accordingly that of the controller. The DHOBE based controller works best for large step disturbances and model errors, which are hard for other methods to compensate for.

The DHOBE algorithm also gives us the freedom of applying the model-reference method or the set-valued method. Because for each recursion, the returned result is not a single model (point) but a set or outer bounding ellipsoid (set) of the estimated parameters. If, according to the algorithm, the center of the ellipsoid each time is taken as the model coefficients, the explicit model update is implemented which leads to a model-reference method. If we choose the worst-case point (point that causes the largest cost function) in the set, then we can apply the Set-Valued method with the help of worst case approach. Both cases are simulated and the results are compared in Chapter 4.

3.1.2 Algorithm

The main idea of DHOBE is recursively obtaining the ellipsoidal outer bounds to the membership set. It can be applied to linear-in-the-parameters models like the Hammerstein model introduced in Chapter 1. And this feature ensures that the algorithm can be used to estimate the parameters of quadratic models. This broadens the scope of the control method to processes in which the Hammerstein model is normally used to approximate the nonlinearity of the process.

Assume that the process model is of the following form:

$$y(t) = \theta^{*T}\Phi(t) + v(t) \tag{3.1}$$

where θ^* is the true parameter vector and $\Phi(t)$ is the input vector. $v(t)$is the noise term which is bounded by γ, i.e

$$|v(t)| \leq \gamma. \tag{3.2}$$

Suppose that at time instance t-1, the membership set of the parameters of the model is outer bounded by the ellipsoid E_{t-1}. It can be defined by its center from $\theta(t-1)$, its orientation from the positive definite matrix $P^{-1}(t-1)$ and its size from the uncertainty parameter $\sigma^2(t-1)$:

$$E_{t-1} = \{\theta \in \Re^N : [\theta - \theta(t-1)]^T P^{-1}(t-1)[\theta - \theta(t-1)] \leq \sigma^2(t-1)\} \tag{3.3}$$

On the other hand, at time instance t, we have an observation of the process y_t which, after defining the noise bound γ, we can utilize to obtain another set S_t as follows:

$$S_t = \{\theta \in \Re^N : [y(t) - \theta^T\Phi(t)]^2 \leq \gamma^2\}. \tag{3.4}$$

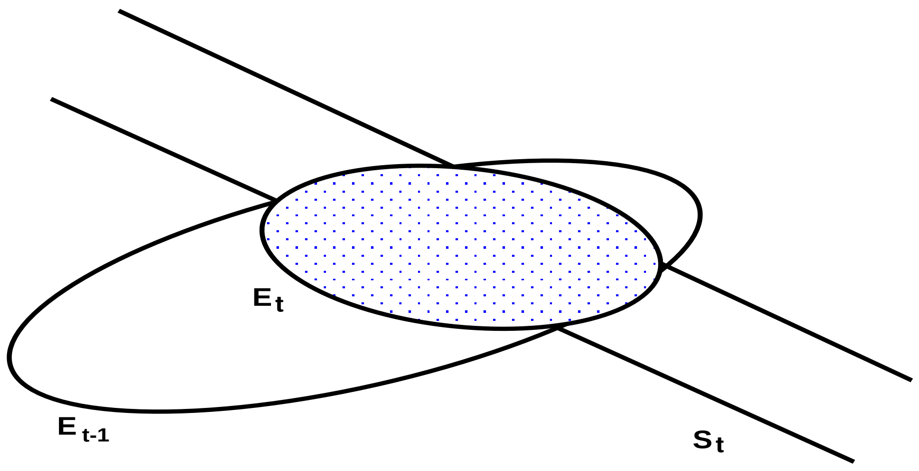

Figure 3.1: Recursive Formation of Bounding Ellipsoid [1]

S_t will intersect with E_{t-1} which will enable us to recursively enclose the intersection of the two sets by another ellipsoid E_t as follows:

$$\begin{aligned} E_t &= \{\theta \in \Re^N : (1-\lambda_t)[\theta - \theta(t-1)]^T P^{-1}(t-1)[\theta - \theta(t-1)] + \lambda_t[y(t) - \theta^T \Phi(t)]^2 \\ &\leq (1-\lambda_t)\sigma^2(t-1) + \lambda_t \gamma^2\}, \end{aligned} \tag{3.5}$$

where the updating gain λ_t is introduced. The gain λ_t is positive and time varying. We can also consider $(1-\lambda_t)$ as the forgetting factor. λ_t is chosen to minimize $\sigma^2(t)$ at each time instance in order to decrease the size of the ellipsoid from run to run as, is easily seen from the following updates of the ellipsoids.The recursive formation of the ellipsoids is illustrated in a 2-dimensional example in Figure 3.1. It is shown in [1] that

$$E_t = \{\theta \in \Re^N : [\theta - \theta(t)]^T P^{-1}(t)[\theta - \theta(t)] \leq \sigma^2(t)\}, \tag{3.6}$$

can be transformed to the following recursive expressions.

$$P^{-1}(t) = (1-\lambda_t)P^{-1}(t-1) + \lambda_t \Phi(t)\Phi^T(t) \tag{3.7}$$

$$\sigma^2(t) = (1-\lambda_t)\sigma^2(t-1) + \lambda_t\gamma^2 - \frac{\lambda_t(1-\lambda_t)[y(t) - \Phi^T(t)\theta(t-1)]^2}{1-\lambda_t+\lambda_t\Phi^T(t)P(t-1)\Phi(t)} \tag{3.8}$$

$$\theta(t) = \theta(t-1) + \lambda_t P(t)\Phi(t)[y(t) - \Phi^T(t)\theta(t-1)]. \tag{3.9}$$

Using the matrix-inversion lemma (3.7) can yield

$$P(t) = \frac{1}{1-\lambda_t}[P(t-1) - \frac{\lambda_t P(t-1)\Phi(t)\Phi^T(t)P(t-1)}{1-\lambda_t+\lambda_t\Phi^T(t)P(t-1)\Phi(t)}]. \tag{3.10}$$

After computing the center of the ellipsoid E_t, we can use it as a point estimation of the parameter vector for the model.

The above updating of model estimation is not performed every time. The following criterion is checked at every run in order to decide whether the updating of the ellipsoid's information should be performed or not. If

$$\sigma^2(t-1) + \delta^2(t) \leq \gamma^2, \tag{3.11}$$

where $\delta(t)$ is the prediction error given by

$$\delta(t) = y(t) - \Phi^T(t)\theta(t-1), \tag{3.12}$$

then there will be no update; all the parameters use the values of the previous run.

Otherwise, λ_t can be calculated according to the following.

$$\lambda_t = \min(\lambda_{max}, \nu_t),$$

where

$$\nu_t = \begin{cases} \lambda_{max} & \text{if} \quad \delta_t^2 = 0 \\ (1-\beta_t)/2 & \text{if} \quad G_t = 1 \\ (1-\sqrt{G_t/(1+\beta_t(G_t-1))})/(1-G_t) & \text{if} \quad \beta_t(G_t-1)+1 > 0 \\ \lambda_{max} & \text{if} \quad \beta_t(G_t-1)+1 \leq 0 \end{cases}$$

where $\lambda_{max} \in (0,1)$ is a user defined upper bound of λ_t, and

$G_t = \Phi_t^T P_{t-1} \Phi_t$

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

3.1.3 Rescue Procedure

The rescue procedure of the algorithm was developed by Rao and Huang in [2]. It is particularly necessary when there is a large jump in the parameters of the model or big disturbances occurred in the process. In such cases, the algorithm might return an empty set and thus there is no bounding ellipsoid generated. This is because when the parameters change abruptly in a big step, the intersection of E_{t-1} and S_t will be void as illustrated in Figure 3.2. At this time the calculation of the σ^2 will become negative which is a result and also an indication that there will be no bounding ellipsoid. This is the failure of the algorithm and the rescue procedure is called at this time to enlarge the size of E_{t-1} so that the intersection of S_t and the enlarged ellipsoid E_{t-1} will no longer be void. This rescue procedure will migrate the center of the ellipsoid to the real parameter vector θ^* which will reduce the parameter estimation error.

3.2 DHOBE Algorithm Implementation

For most semiconductor manufacturing processes, the model can be summarized as a linear-in-the-parameters form as follows. This form includes the second order and interaction terms but still linear in the parameters (coefficients),

$$y_t = u^T \Theta + \eta_t$$

where

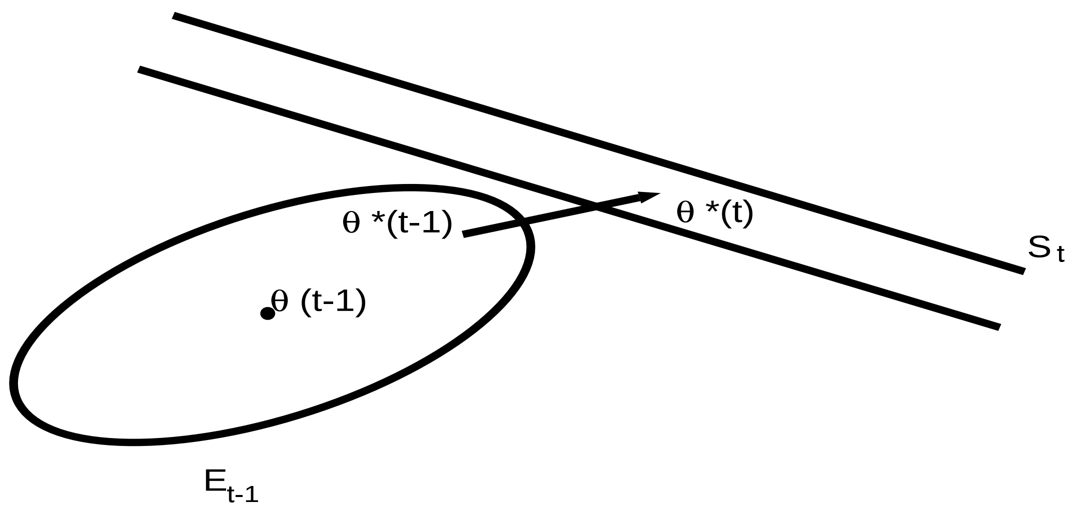

Figure 3.2: DHOBE Rescue Procedure Illustration [1]

u is the vector of controllable inputs

θ is the parameter of the model

η_t is the noise term.

The DHOBE algorithm can be applied to such models by producing a set of estimation of the parameters Θ which is bounded by an ellipsoid defined as:

$$E_\Theta = \{\Theta_t : (\Theta_t - \Theta_t^{(+)})^T P_t^{-1} (\Theta_t - \Theta_t^{(+)}) \leq \sigma_t^2\},$$

where

P_t^{-1} is a positive definite matrix with dimension the same as the parameter vector. It describes the shape of the ellipsoid.

$\Theta_t^{(+)}$ is the center of the ellipsoid and is considered as the current estimation of the parameters.

σ_t^2 is the uncertainty of the estimated parameters and defines the size of the ellipsoid.

All the above parameters are updated at each run and are used to estimate the model parameters for the next run. As stated before, the only knowledge required for the process is the strict bound of the noise η_t. i.e.

$$|\eta_t| < \gamma.$$

This is also the sufficient condition to ensure convergence.

The procedures used in the DHOBE algorithm are listed below:

Step 1) At each run t, calculate the error residual:

$$\delta_t = y_t - u_t^T \Theta_{t-1}^{(+)}$$

where

y_t is the measurement made(or the process response) at this run.

u_t is the recipe calculated for this run.

$\Theta_{t-1}^{(+)}$ is the model parameters updated at the last run.

δ_t is the error.

Step 2) Check if the following inequality holds

$$\gamma^2 \geq \sigma_{t-1}^2 + \delta_t^2.$$

If Yes, then it is known that the process response is still in the acceptable range. i.e. the measurement is redundant and contains no new information for updating the model coefficients. Return to Step 1) without calculating any parameters of the algorithm. The parameters for the model also remain the same as in the previous run. It is this check that saves a lot of computational load and makes the algorithm efficient when only small drift exist.

If No, the algorithm does the following steps to update the parameters within itself and also updates the coefficients of the model.

Step 3) Compute two intermediate scalar variables:

3a)$G_t = u_t^T P_{t-1} u_t$

3b)$\beta_t = (\gamma^2 - \sigma_{t-1}^2)/\delta_t^2.$

Step 4) Compute an intermediate variable ν_t and the update factor λ_t

$\lambda_t = \min(\lambda_{max}, \nu_t)$

$$\nu_t = \begin{cases} \lambda_{max} & \text{if} \quad \delta_t^2 = 0 \\ (1-\beta_t)/2 & \text{if} \quad G_t = 1 \\ (1-\sqrt{G_t/(1+\beta_t(G_t-1))})/(1-G_t) & \text{if} \quad \beta_t(G_t-1)+1 > 0 \\ \lambda_{max} & \text{if} \quad \beta_t(G_t-1)+1 \le 0 \end{cases}$$

λ_t is the design factor and is in the range of $[0, 1)$. $1-\lambda_t$ is the forgetting factor.

Step 5) Update the parameter uncertainty factor

$\sigma_t^2 = (1-\lambda_t)\sigma_{t-1}^2 + \lambda_t\gamma^2 - \lambda_t(1-\lambda_t)\delta_t^2/(1-\lambda_t+\lambda_t G_t)$

Step 6) Check if $\sigma_t^2 > 0$,

If Yes, proceed to step 7.

If No, do the following:

6a) Compute the intermediate scalar variable

$$\kappa = \begin{cases} \delta_t^2 + \gamma^2 - 2\gamma|\delta_t| & \text{if} \quad \lambda_t \ne \lambda_{max} \\ \lambda_{max}\left(\frac{\delta_t^2}{1-\lambda_{max}+\lambda_{max}G_t} - \frac{\gamma^2}{1-\lambda_{max}}\right) & \text{if} \quad \lambda_t = \lambda_{max} \end{cases}$$

6b) Reset the uncertainty parameter for time t-1

$\sigma_{t-1}^2 = \kappa + \zeta$

6c) Return to Step 3b).

Steps 6a)-6c) is the rescue procedure for the case when σ_t^2 is negative and the returned bounding ellipsoid is an empty set. ζ is the user specified inflation parameter. It will inflate the collapsed ellipsoid sufficiently in order to contain the new $\Theta^{(+)}$.

Step 7) Update the ellipsoid's structure matrix

$$P_t^{-1} = (1 - \lambda_t)P_{t-1}^{-1} + \lambda_t u_t u_t^T$$

. Step 8) Update the auxiliary recursive-least-square matrix

$$P_t = \frac{1}{1-\lambda_t}[P_{t-1} - \frac{\lambda_t P_{t-1} u_t u_t^T P_t - 1}{1 - \lambda_t + \lambda_t G_t}]$$

Step 9) Update the new center of the ellipsoid

$$\Theta_t^{(+)} = \Theta_{t-1}^{(+)} + \lambda_t P_t u_t \delta_t$$

and return to Step 1).

3.3 RbR Controller Based on DHOBE

3.3.1 Model-Reference RbR Controller Based on DHOBE

The DHOBE algorithm is applied to two forms of the RbR controller. The first one is the model reference (or internal model controller discussed in Section 2) and the second form is used as the Set-Valued controller with the worst case approach. These two methods of implementation of the algorithm are discussed in this section.

In model reference RbR controller, DHOBE is used as the algorithm to implement the run-by-run model parameter estimation. The recipes are generated using the minimization of the squared error between the model's predictive output and the target value. Figure 3.3 illustrates the basic flow chart of the controller. The first step is to set the parameters related to the internal process model. This includes the following:

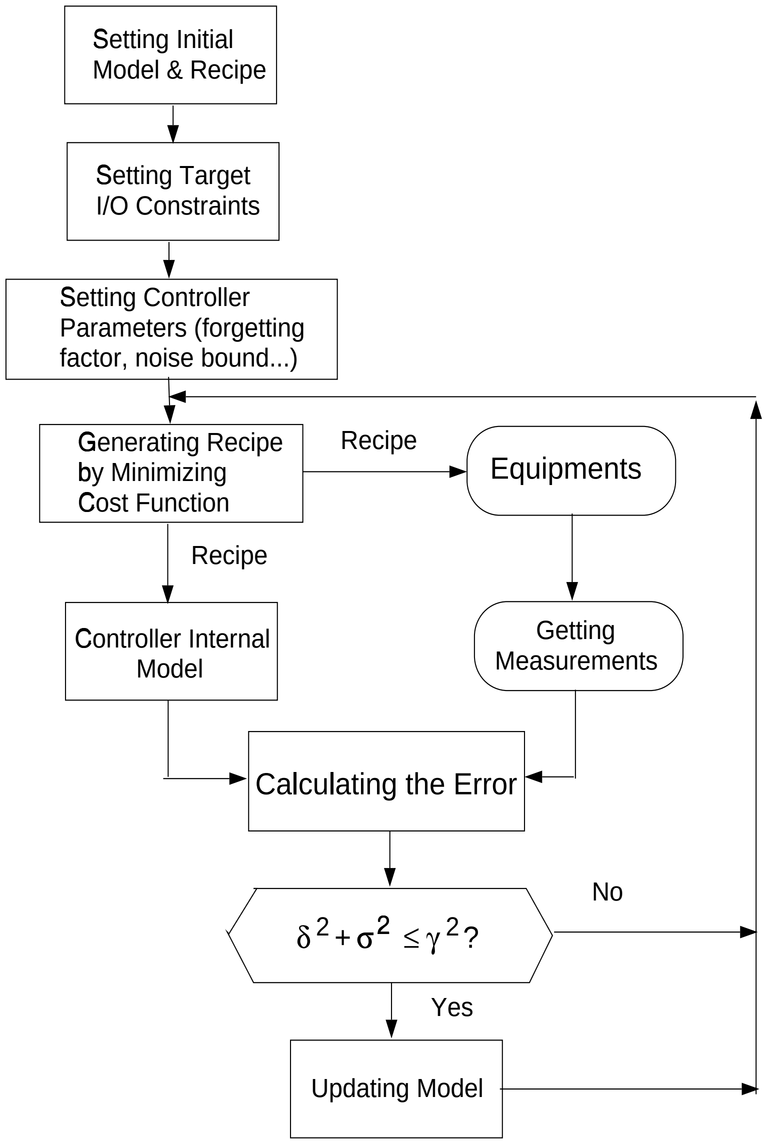
Setting Initial Model & Recipe
Setting Target I/O Constraints
Setting Controller Parameters (forgetting factor, noise bound...)
Generating Recipe by Minimizing Cost Function
Recipe
Equipments
Recipe
Controller Internal Model
Getting Measurements
Calculating the Error
$\delta^2 + \sigma^2 \leq \gamma^2$?
No
Yes
Updating Model

Figure 3.3: Block Diagram for DHOBE Based Controller

- Building the initial model which might be obtained from off-line experiments.
- Setting the target value of the process outputs.
- Setting the constraints of the inputs.
- Setting the initial recipes of the equipment which might be the nominal value of the inputs.

The second step is to set parameters related to the controllers. This parameter setting is not easy as it is equivalent to the tuning of the controllers. But after some off-line simulations are completed, the recommended values can be available. These parameters include:

- Setting the strict noise bound γ.
- Setting the updating factor's (λ) upperbound λ_{max}.
- Setting the rescue step length κ which is normally set to 1.
- Setting the ellipsoid's orientation matrix P which is normally set to I at the start.

The choice of the noise bound is most important. When γ is chosen to be larger than the actual bound, the tracking ability of the algorithm will be increased. But from the DHOBE algorithm, it is easy to see that when the noise bound is selected too large, the update of the model will not happen until the model's output significantly deviates from the target. This is not expected especially when the process is undergoing small drifts. The sensitivity of the controller will be decreased because of this larger noise bound. In our simulations, the process noise and measurement noise are combined to a white noise and the normal distribution has zero mean and variance σ. We took 3σ as the noise bound and it is shown that this bound is appropriate enough for the simulated circumstances.

The setting of the maximum value of the updating factor (λ_{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 (σ^2) may change so rapidly that the ellipsoid's boundary excludes the real process parameter (θ^*). If θ^* ends up too far outside the ellipsoid, the result is usually an ellipsoid which collapses to the empty set and invokes the rescue procedure. The re-inflated ellipsoid can then miss θ^* again at the next update cycle and so on resulting in a non-converging oscillation of the ellipsoid. In our simulations, we use 0.4 as the maximum value and by selecting the proper re-inflation step κ, satisfactory control results can be achieved.

After having a model estimation for each run, the recipe can be calculated for the next run by minimizing a predefined cost function. The cost function normally takes the form of squared error between target value and the model's output using the current model estimation. When there is only one response the form might be

$$\min_u (T - \theta^T u).$$

If there are multiple responses, the cost function might take the form

$$\min_u (T - \theta^T u)^T W (T - \theta^T u),$$

where W is the weight diagonal matrix assigned for each response. The value of each element represents the priority of each response. The most important

output is assigned the largest weight, which is used in order to keep the output value as close to its target value as possible. T is the target vector.
If the change of recipes incurs large costs, then this change should also be taken into consideration by adding an additional term into the cost function as in [5]:

$$\min_{u_t}\{w_1(T-\theta^T u_t)+w_2(u_t-u_{t-1})^T\Gamma(u_t-u_{t-1})\},$$

where Γ is the weight diagonal matrix assigned to each input of the recipe. If the cost for changing a certain tunable input is higher, then we can assign larger weight to that particular input so that by optimizing the cost function, it is kept as close to the previous run's value as possible. w_i are the weight terms also.

The minimization is achieved by finding the optimal recipe during each run through line searching. The calculated inputs should also satisfy the constraints that are essential for equipment and process requirements. These constraints are normally in the form of ranges which are quite natural and easy to get. These constraints guarantee that the optimal recipe is feasible physically and reasonable to the equipment.

After the recipe is calculated, the equipment tunable inputs are adjusted accordingly, which might cause the output of the real process to change. This change is reflected in the characteristic of the product. Measurements are conducted and fed back to the controller as new information. Then according to DHOBE, the residual error $\delta(t)$ between the current measurement and the model's output (the current recipe and the previous run's model) will be calculated and used to estimate the next run's model.

At this time, the controller will decide if the updating is necessary by the following criterion:

$$\delta^2 + \sigma^2 \leq \gamma^2$$

If the inequality is satisfied, the controller uses the previous model for the next run. And the cycle restarts.

3.3.2 Set-Valued RbR Controller Based on DHOBE

The Set-Valued RbR Controller using DHOBE is similar to the model-reference one except for the part of optimization of the cost function and generation of recipes. The model updating still uses the centers of the ellipsoids as the model parameter estimates, recursively. But after the outer bounds of the ellipsoids are generated, the optimization selects the model from these ellipsoids (sets) and tries to find the min-max of the cost function. Using the results from [7] we get the following.

Suppose at run t, the ellipsoid is generated by DHOBE with the parameters: center of ellipsoid θ_t^*, orientation P_t and size σ_t^2. The optimization of the cost function is implemented as

$$\min_{u \in U} \max_{\underline{y}_t \leq y \leq \bar{y}_t} l(y)$$

where U is the feasible set of inputs; this can be achieved by imposing constraints according to the requirements from the process.

$$\underline{y}_t = \theta_t^{*T} u - \sqrt{u^T \frac{P_t}{\sigma_t^2} u} - \sqrt{g}$$

$$\bar{y}_t = \theta_t^{*T} u + \sqrt{u^T \frac{P_t}{\sigma_t^2} u} + \sqrt{g}.$$

$\underline{y}_t$ and $\bar{y}_t$ define the inner maximization scope. In order to simplify the calculation, we impose the restriction that the cost function is convex. Then the maximization will only appear on either of the two points.

For multiple responses process, just like the former controllers, we used the Pareto-optimal method so that different weights are selected for different responses and then sum them together. The worst case treatment is to select the larger one of the cost functions between the upper and lower bounds of the response among all the responses and sum the weighted sub-functions together.

Chapter 4

Simulations and Comparisons

4.1 Simulation Set-up

The simulations were performed using MATLAB version 5 both for real process outputs and the controller algorithm. First, the process was simulated using the model that was summarized from experiments. Its output is considered as the real process output. The model is in polynomial form and can be linear, quadratic or of any higher order. This model is used as the real process which is subject to noise, drift, large shifting and all the disturbances that can happen. Then the calculated recipes are input to the model and finally output measurements are obtained. These measurements are also simulated as corrupted by noise.

In the simulation, the noises for processes and measurements are combined as one. Furthermore the assumption was made that the noises are normally distributed with definite variance and zero mean. The noises to different responses of the same process are not correlated. This assumption does not affect the DHOBE method as the algorithm only requires the knowledge about the bounds of the noise and no prior knowledge or assumption about the noise distribution is required. Other forms of noises are also simulated and the results are not different from the normally distributed one.

The slow drifting is simulated by adding a small positive or negative value to the output of the process. This may also be achieved by slightly drifting the parameters of the model. Actually, they have the same effect on the controller, and in the simulation the first method was used. This alteration also applies to large abrupt shifting.

In MATLAB, the optimizing function with constraints: "constr" serves as the core of the optimization. It is used to find the optimized recipes that minimize the cost function, either in DHOBE-MR (Model Reference version) or DHOBE-SV (Set-Valued version). The function uses several inputs to implement the optimization: u_0 is the starting point for searching. ulb and uub are the lower and upper bounds of the input u. These bounds were used as the constraints defined by equipment and process requirements. The calculated recipe is restricted to this range.

The model embedded into the controller can be assumed as perfect model; i.e. it reflects the real process models with the same parameters and structures. In the simulations, this means that the initial model in the controller is the same as the process model. But in practice, the real process model is unknown. The controller's model is only an approximation to the process based on the past experimental data. No matter how close this approximation is, the internal model will for sure have model errors . This case was also simulated by changing the parameters and structures in a certain range. The simulation shows that the imperfect models only affect the starting transition phase. For both kinds of DHOBE controllers, this transition period is rather short: it only takes one to two runs. After each simulation, statistical performance analysis was performed and the controlled outputs were compared with the uncontrolled one graphically.

4.2 Performance Measures of RbR Controllers

There are different ways of evaluating a RbR controller against its peers. They often lead to variation analysis of the sampling data from the experiments. This was also simulated using the MATLAB program. However, a general criterion for the RbR controller, actually many other controllers, is stated as follows.

1. The controller's ability to track the target value set before hand without lag. This is especially true for the semiconductor manufacturing industry considering the cost of each lot of wafer and all the money and time consuming manufacturing processes before.

2. The controller's ability to prevent disturbances from influencing the process outputs.

3. The controller's ability to reject noise. i.e. not to respond to spurious fluctuations.

Sometimes increased product quality is associated with improved process performance, such as process capability, or C_{pk}, defined as:

$$C_{pk} = \min(C_{pl}, C_{pu})$$

where

$$C_{pl} = |m - L|/3\sigma$$

$$C_{pu} = |m - U|/3\sigma$$

m is the mean, σ is the standard deviation, U and L are the upper and lower limits.

Another performance measure that indicates the frequency of recipe changes is the average adjustment interval (AAI)

In the comparisons of the simulation results between the DHOBE controller and other RbR controllers, the following statistical measures are used to evaluate the performance of the control action.

- $\bar{y}_i$ is the mean of the sampling values from the real process ith output.
- S_{y_i} is the standard deviation of the process ith output.
- $MSD(y_i - T_i)$ is the square root mean square deviation of the process ith output from its target value.

Sometimes, when the issue of changing the input parameters is also taken into consideration, the standard deviation for each input will be considered also.

4.3 Simulation Results and Analysis

4.3.1 Simulations Comparison with EWMA

For the CMP Model in this simulation, the following linear model is used and is considered as the perfect model for the real process. The EWMA method is used as the comparison. The original model is for multiple objective purposes: to optimize both Removal Rate and Non-Uniformity. In [6], only one response removal rate was simulated. Both algorithms are simulated under the same circumstances: i.e. the same noise, drifting, disturbance, model and model error. Standard deviation and MSE (Mean Square Error) are calculated and taken as the main metric for evaluation.

Process Models

The process model introduced in Chapter 1 is used here for both the real process model and the internal model of the controller. No error in either coefficients or structure when summarizing the model is considered.

$$y[n] = -1382.60 + [50.18, -6.65, 163.4, 8.45] \times u^T(n) + \omega[n] + \delta[n].$$

The simulation results are illustrated in Figure 4.1 and Figure 4.2 for DHOBE-MR and DHOBE-SV.

From Table 4.1, it can be seen that the compensation effect for DHOBE and EWMA have no big difference as measured by the MSE and standard deviation for the simulated number of runs. The weight of the EWMA controller is selected as 0.6 in this case.

But when there is some model error, which is common in real applications, the DHOBE-MR and DHOBE-SV have faster convergent characteristic (as seen from simulations).

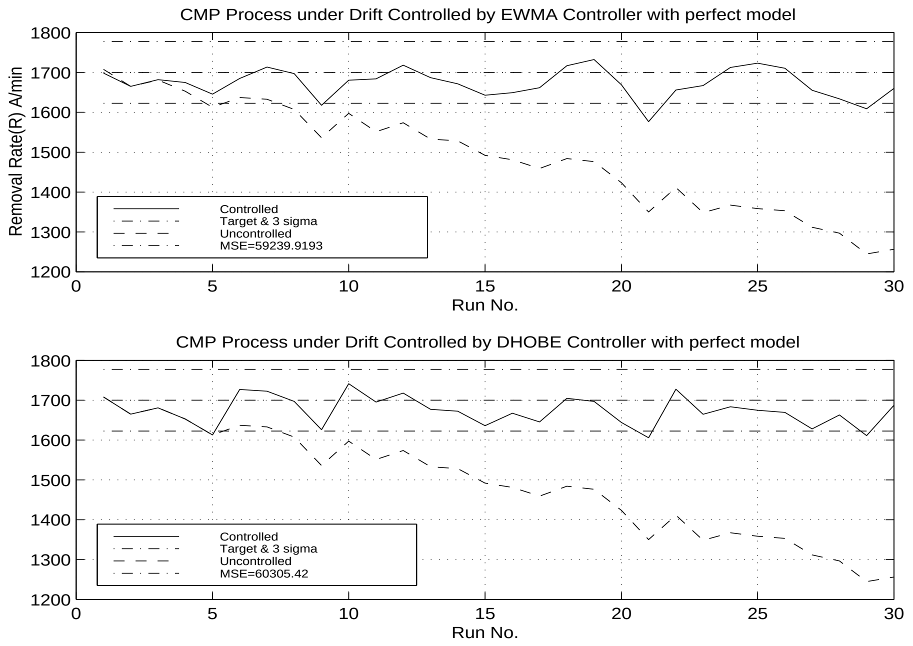

Figure 4.1: Comparison of DHOBE-MR and EWMA with Linear Perfect Model under Drifting

Method	$\bar{y}$	S_y	MSD
EWMA	1674.4	36.7	44.8
DHOBE-MR	1664.6	36.9	51.5
DHOBE-SV	1669.9	35.1	46.4

Table 4.1: EWMA and DHOBE Performance for Linear Perfect CMP Model

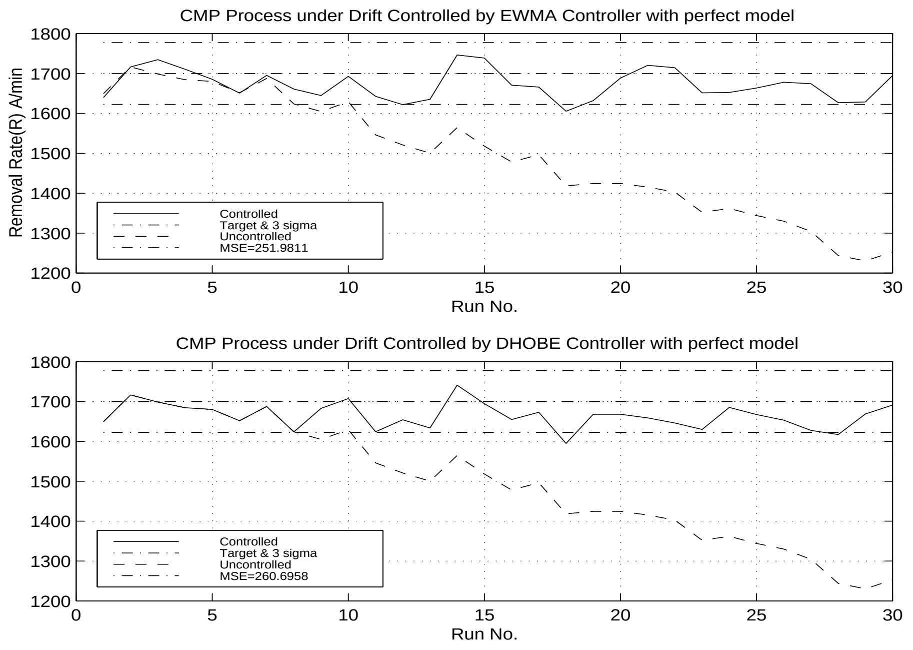

Figure 4.2: Comparison of DHOBE-SV and EWMA with Linear Perfect Model under Drifting

The simulation is somewhat rough for this case as the process model was taken as 80% of each parameters of the real process. DHOBE-MR and DHOBE-SV are shown to track the target much faster than EWMA under this circumstance. This is also illustrated in Table 4.2 by considering the STD and MSD. The simulation of EWMA controller used 0.3 as the weight.

Simulations were also performed for the case when there is a large step disturbance during the operation. The simulation result also shows that DHOBE-MR and DHOBE-SV are better than EWMA because EWMA is suitable for gradual mode. It is hard for it to compensate for large variance in several runs. The step disturbance in the simulation was experimented by changing the model parameters of the real process greatly. The resulted shifting value equals to the target

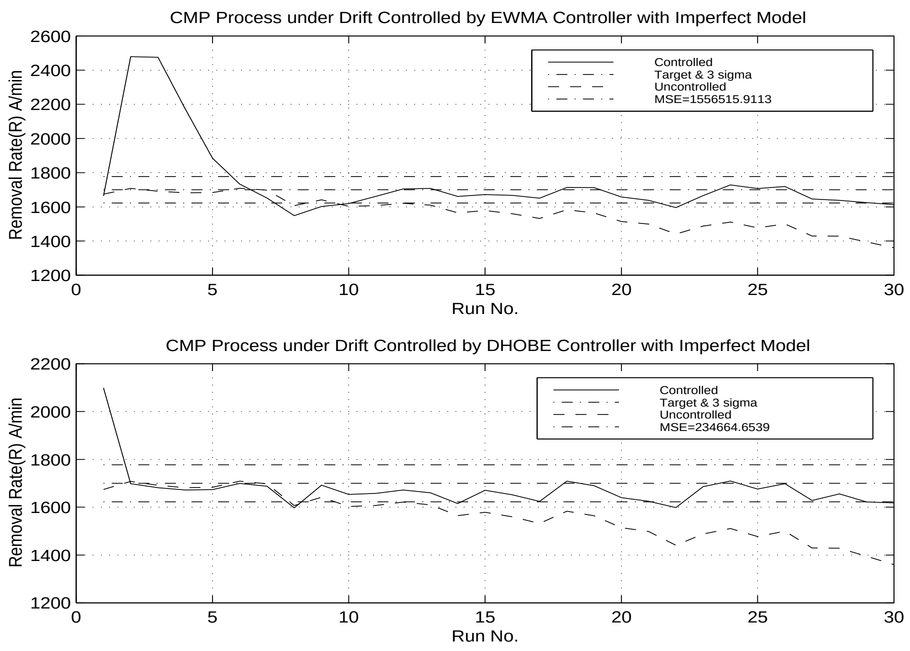

Figure 4.3: Comparison of DHOBE-MR and EWMA for Linear Imperfect Model under Drifting

Method	$\bar{y}$	S_y	MSD
EWMA	1742.9	219.7	223.9
DHOBE-MR	1672.5	90.9	95.1
DHOBE-SV	1685.7	90.0	91.3

Table 4.2: EWMA and DHOBE Performance for Linear Imperfect CMP Model

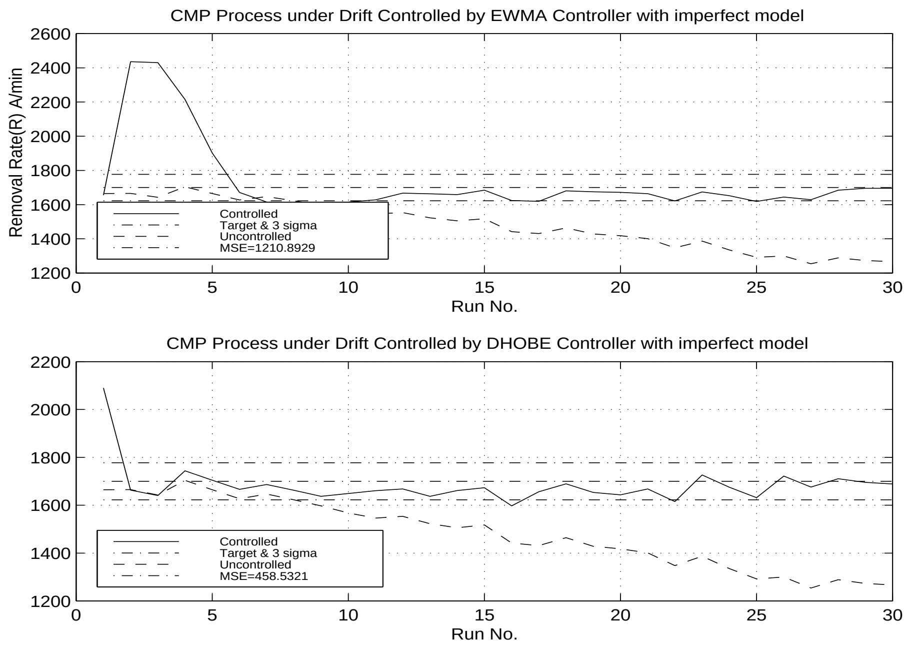

Figure 4.4: Comparison of DHOBE-SV and EWMA for Linear Imperfect Model under Drifting

value + 350. The weight for EWMA controller is selected as 0.43.

Bad Data case was simulated and the result for DHOBE-MR and DHOBE-SV are not better in view of the MSE. But after one run of large deviation from the target, the response returned to the acceptable range (3σ) very quickly. The bad data used in the simulation is 500 above target value. In the simulation, the weight for EWMA controller is selected as 0.6.

4.3.2 Simulations Comparison with OAQC

The simulation setup and result for OAQC can be found in [5]. It does not provide much detailed information about the algorithm. Instead the process model for CMP and the simulation data were provided in detail. Simulations using

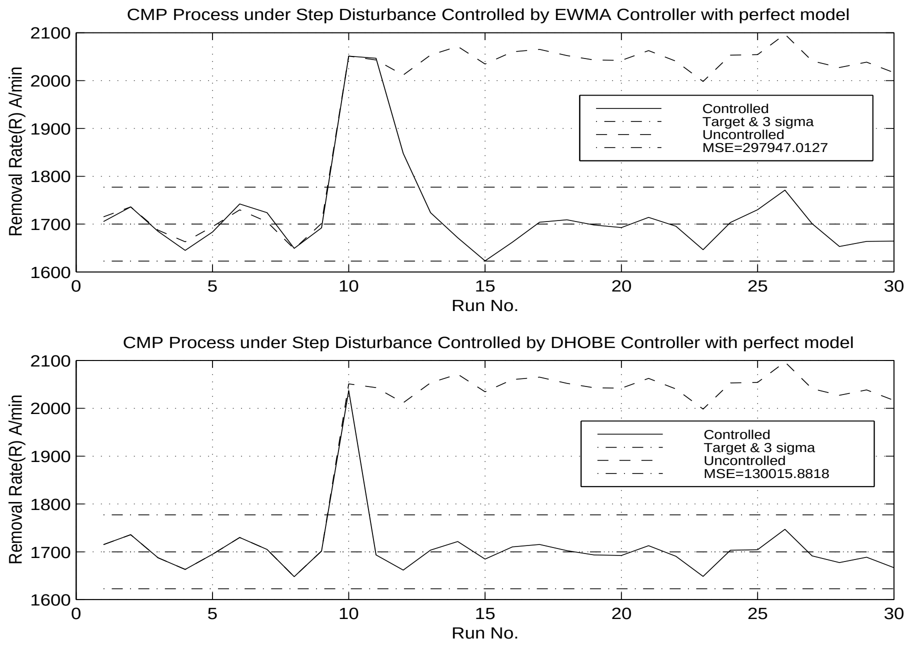

Figure 4.5: Comparison of DHOBE-MR and EWMA for Linear Perfect Model under Step Disturbance

Method	$\bar{y}$	S_y	MSD
EWMA	1717.1	97.9	100.8
DHOBE-MR	1708.9	67.0	68.1
DHOBE-SV	1710.9	68.6	69.8

Table 4.3: EWMA and DHOBE Performance for CMP Model w/Step Disturbance

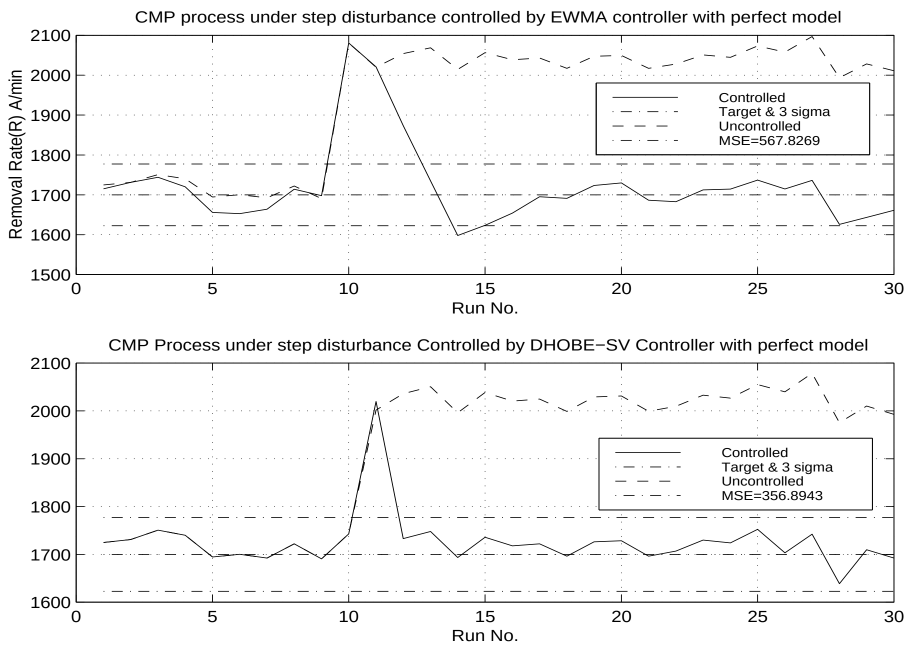

Figure 4.6: Comparison of DHOBE-SV and EWMA for Linear Perfect Model under Step Disturbance

Method	$\bar{y}$	S_y	MSD
EWMA	1699.8	106.0	106.0
DHOBE-MR	1693.1	128.2	128.5
DHOBE-SV	1689.9	125.1	125.6

Table 4.4: EWMA and DHOBE Performance for CMP Model w/ Bad Data

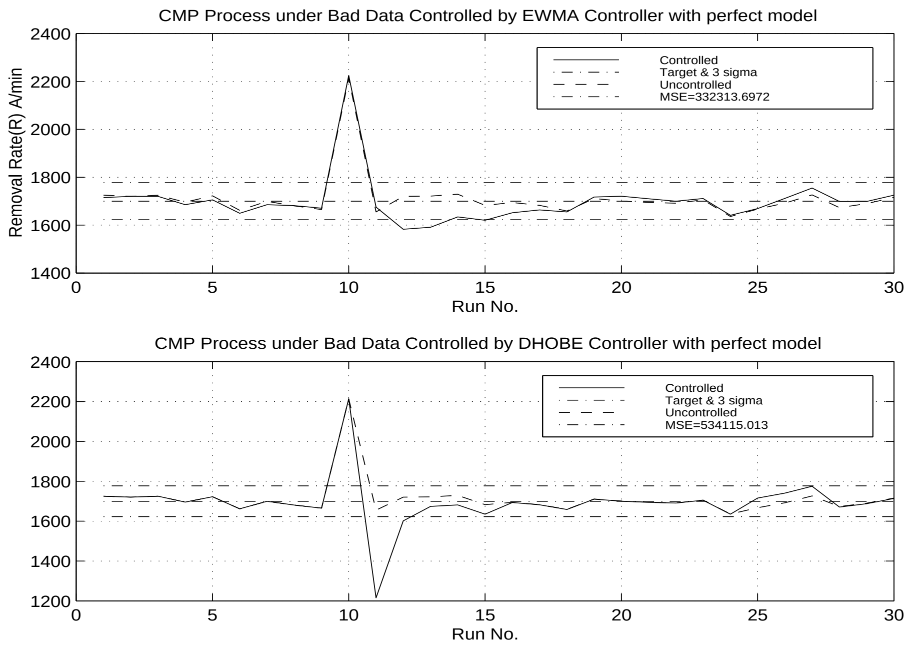

Figure 4.7: Comparison of DHOBE-MR and EWMA for Linear Perfect Model under Bad Data

the model and DHOBE algorithm were fully implemented and the result was compared against OAQC.

The process models obtained from experiments are considered as real process models with drifts and the approximated models are in almost-linear, fully quadratic and linear form. Performance was evaluated according to different kinds of approximate models. This provides us an opportunity to test the DHOBE's robustness to model error. There are two forms of the model with regards to the number of parameters to be tuned: CMP4x2 and CMP3x2.

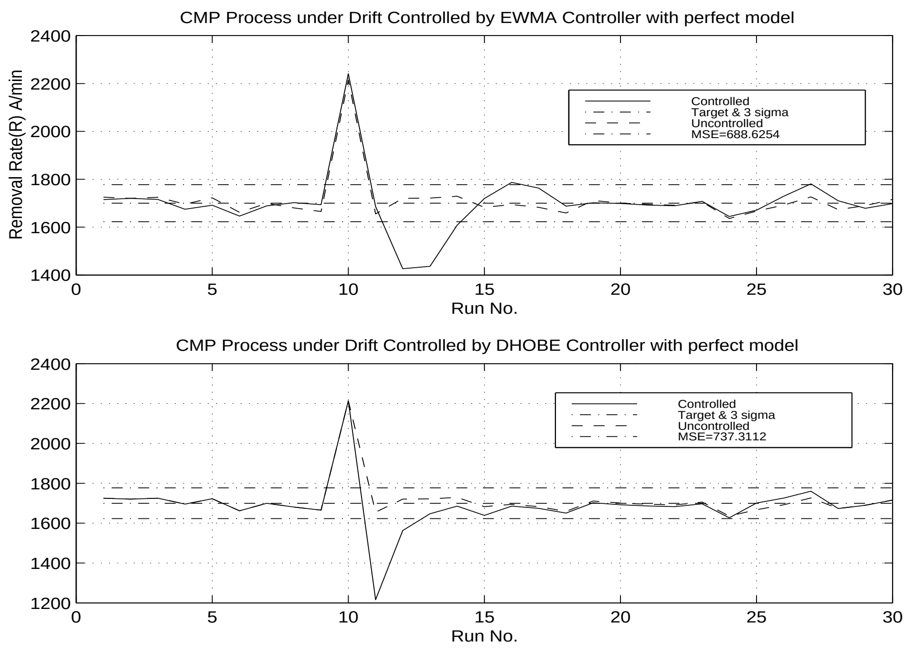

Figure 4.8: Comparison of DHOBE-SV and EWMA for CMP Model w/Bad Data

CMP 4 $\times$ 2 Model

1) Process Model Considered as Real

$$\begin{aligned} y_1 &= 1563.5 + 159.3u_1 - 38.2u_2 + 178.9u_3 + 24.9u_4 - 67.2u_1u_2 - 46.2u_1^2 \\ &\quad -19.2u_2^2 - 28.9u_3^2 - 12u_1t' + 116u_4t' - 50.4t' + 20.4t'^2 + \epsilon_{1,t} \end{aligned}$$

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

where

$t' = (t-53)/53, \epsilon_{1,t} \sim N(0, 60^2), \epsilon_{2,t} \sim N(0, 30^2)$

This model was summarized from 209 wafer experiments and was considered to be the real process model in the simulations. It is a rather complex form as it includes both the quadratic and 2 factor interaction terms model.

y_1 is the removal rate; target value 2000.

y_2 is the with-in wafer nonuniformity. target value 100.

u_1 is the platen speed.

u_2 is the back pressure.

u_3 is the polishing downforce.

u_4 is the profile.

All controllable factors are scaled to [-1,1] range. The target values for y_1 and y_2, 2000 and 100 are unrealistic. These values are set in order to evaluate the performance of the algorithm. For y_1, the larger value the better, and for y_2, the smaller the value the better; and for most of the time these target values cannot be reached.

2)Approximate Initial Models – Quadratic Models (Scenario 1)

First, the fully quadratic model used in the controller is of the following form:

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

and

$$y_2 = 250 + 30u_1 + 100u_2 + 20u_3 + 35u_4 - 30u_1u_2 + 0.05t$$

We can see that there is a big model error comparing with the real process model for both parameters and drifting. The performance measurement is implemented in 5).

3)Approximate Initial Models – Linear Models (Scenario 2)

If the quadratic and interaction terms are dropped then the above model is changed to a linear model. As the real process model is not severely nonlinear, the control effect with the linear model is good. The constraints for the input and output are the same as in the quadratic model.

$$y_1 = 1600 + 150u_1 - 40u_2 + 180u_3 + 25u_4 - 0.9t$$

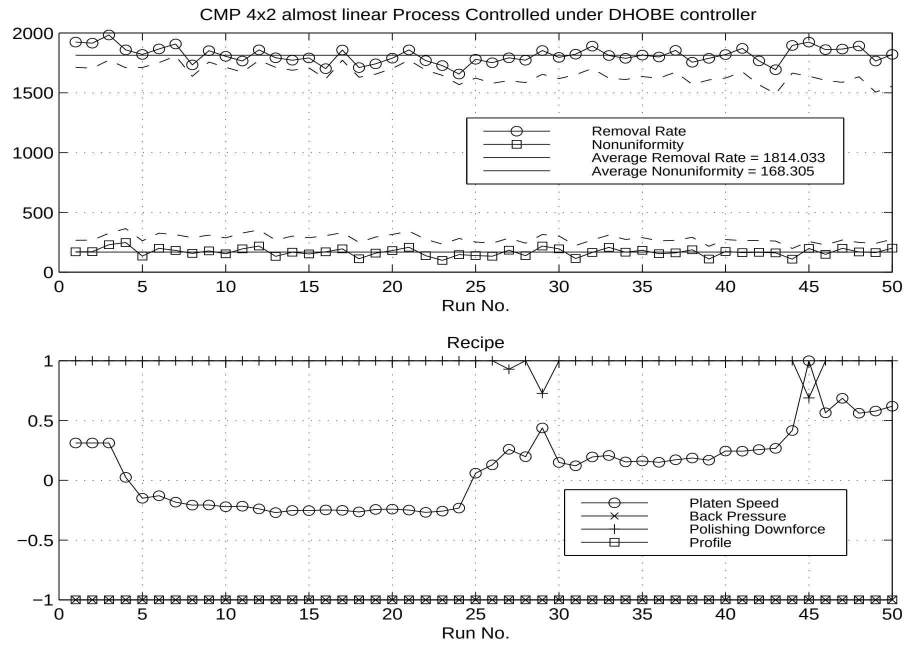

Figure 4.9: CMP4x2 Scenario 1 Controlled by DHOBE-MR

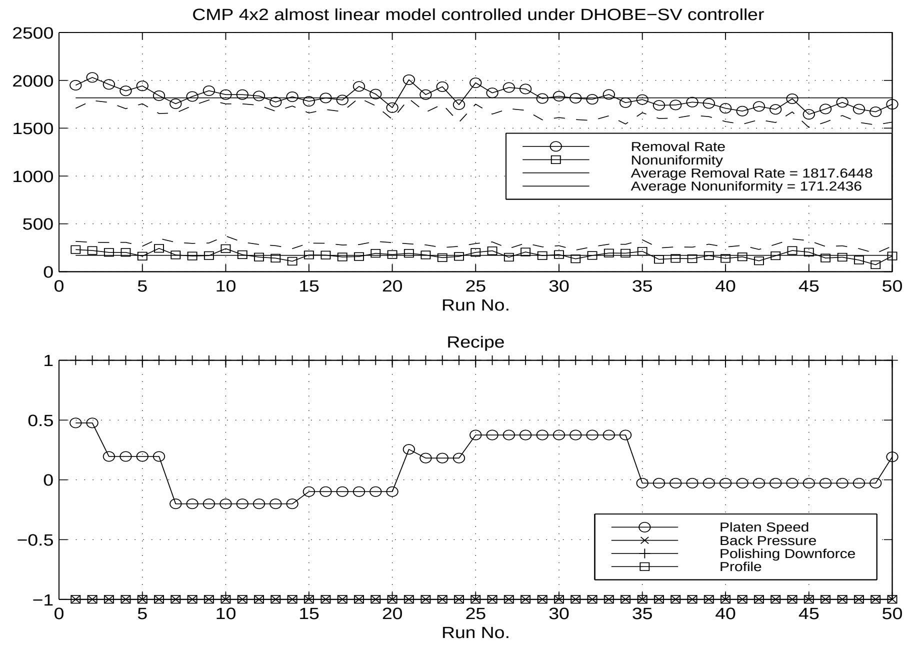

Figure 4.10: CMP4x2 Scenario 1 Controlled by DHOBE-SV

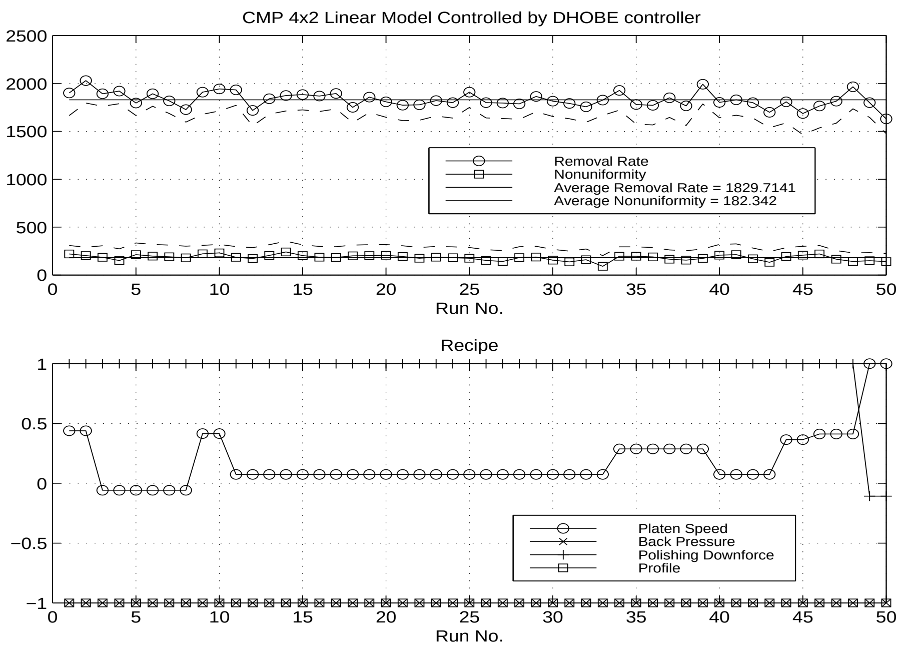

Figure 4.11: CMP4x2 Scenario 2 Controlled by DHOBE-MR

$y_2 = 250 + 30u_1 + 100u_2 + 30u_3 + 35u_4 + 0.05t$

4) Quadratic Models with Step Disturbance (Scenario 3)

The DHOBE algorithm was also tested when abrupt disturbances happened. In this case the quadratic initial model was used and the constraints are the same as before. The abrupt shift to the first response happened at $t = 20$ with magnitude -100 and for the second response the shift happened at $t = 30$ with magnitude 50.

5) Performance Analysis

The OAQC was simulated in [5] under exactly the same circumtances for the above 3 scenarios for 20 times each. The final results with regards to the statistical variance analysis were listed. The DHOBE-MR and DHOBE-SV methods were also tested for 20 times each and using the same kind of performance mea-

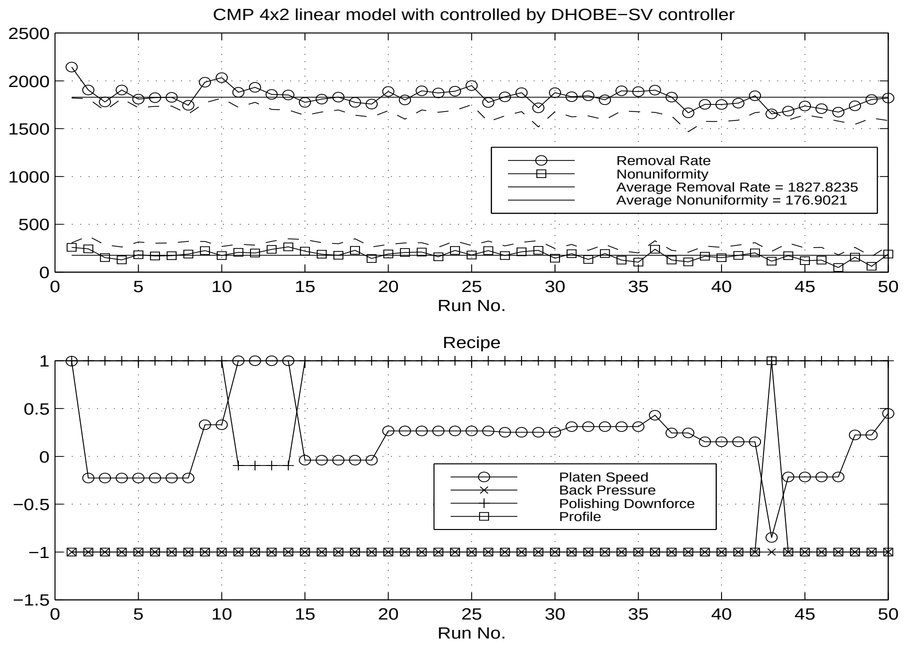

Figure 4.12: CMP4x2 Scenario 2 Controlled by DHOBE-SV

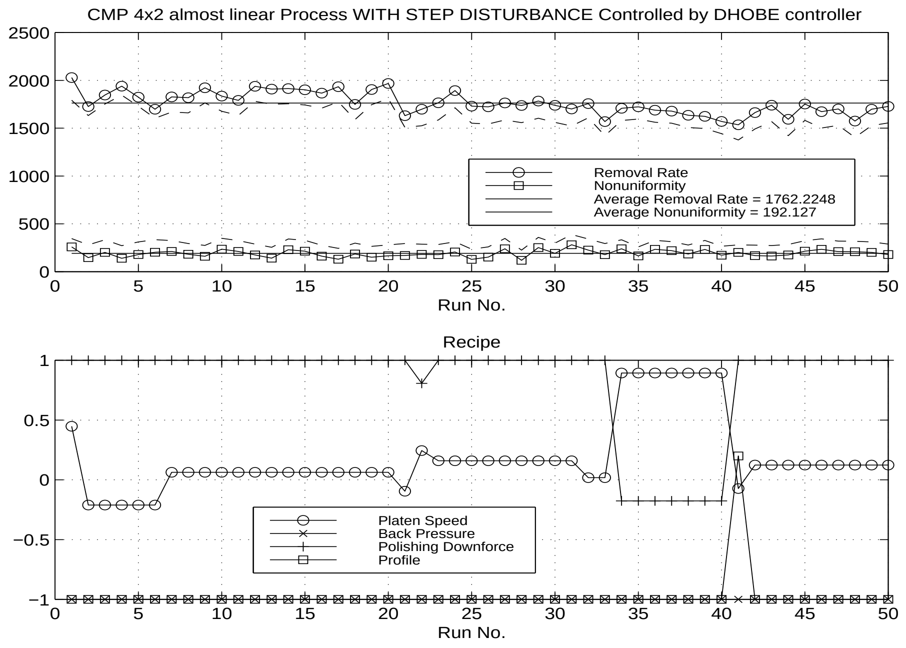

Figure 4.13: CMP4x2 Scenario 3 Controlled by DHOBE-MR

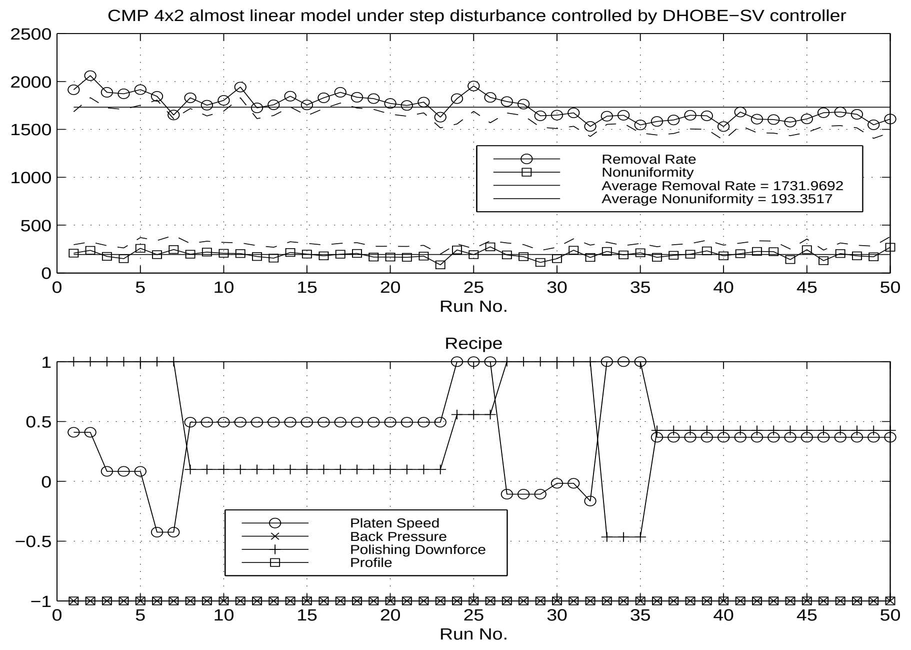

Figure 4.14: CMP4x2 Scenario 3 Controlled by DHOBE-SV

sures; the results are listed in Table 4.5 for comparison. There might be some factors from the tuning of the controller but we can see the tendency, and thus prove the feasibility of the new controller.

Table 4.5 shows that for response 1, the mean value is normally better for DHOBE but the standard deviation is larger than OAQC. For the second response, the mean value is larger but the standard deviation is better for DHOBE.

CMP Nonlinear 3 $\times$ 2 Model

In this section, the second CMP process model is used. It has only 3 controllable factors and the responses are removal rate (y_1) and within-wafer standard deviation (y_2).

1) Real Process Model

$$y_1 = 276.5 + 574.6u_1 + 616.3u_2 - 126.7u_3 - 1109.5u_1^2 - 286.1u_2^2 + 989.1u_3^2$$

Scenario	Method	$\bar{y}_1$	$\bar{y}_2$	S_{y1}	S_{y2}	MSD_1	MSD_2
1	OAQC	1719.7	168.4	70.4	40.1	288.9	79.2
	DHOBE-MR	1754.7	157.3	84.5	35.0	259.7	67.5
	DHOBE-SV	1787.7	168.1	82.8	34.7	228.2	76.9
2	OAQC	1718.2	165.7	72.1	42.0	291.0	78.2
	DHOBE-MR	1781.9	165.0	84.5	36.1	234.2	74.8
	DHOBE-SV	1807.4	177.5	85.9	36.1	211.9	86.1
3	OAQC	1661.2	189.2	89.2	43.5	350.2	99.2
	DHOBE-MR	1741.4	189.1	108.7	35.6	280.8	96.0
	DHOBE-SV	1747.0	190.8	109.2	37.5	275.9	98.3

Table 4.5: OAQC and DHOBE Performance for CMP 4x2 Models

$$-52.9u_1u_2 - 156.9u_1u_3 - 550.3u_2u_3 - 10t + \epsilon_{1,t}$$

and

$$\begin{aligned} y_1 &= 746.3 + 62.3u_1 + 128.6u_2 - 152.1u_3 - 289.7u_1^2 - 32.1u_2^2 + 237.7u_3^2 \\ &\quad -28.9u_1u_2 - 122.1u_1u_3 - 140.6u_2u_3 + 1.5t + \epsilon_{2,t} \end{aligned}$$

where

$\epsilon_{1,t} \sim N(0, 60^2), \epsilon_{2,t} \sim N(0, 30^2)$.

Controllable factors are back pressure downforce (u_1), platen speed (u_2) and slurry concentration (u_3). They are all scaled to [-1,1] range and the target values for y_1 and y_2 are 2200 and 400 respectively. These models are fitted to the results of a 32-wafer experimental design and they served as real process model in the simulation. They are hard to control using a linear model controller as they contain large second-order coefficients. This model also shows the necessity

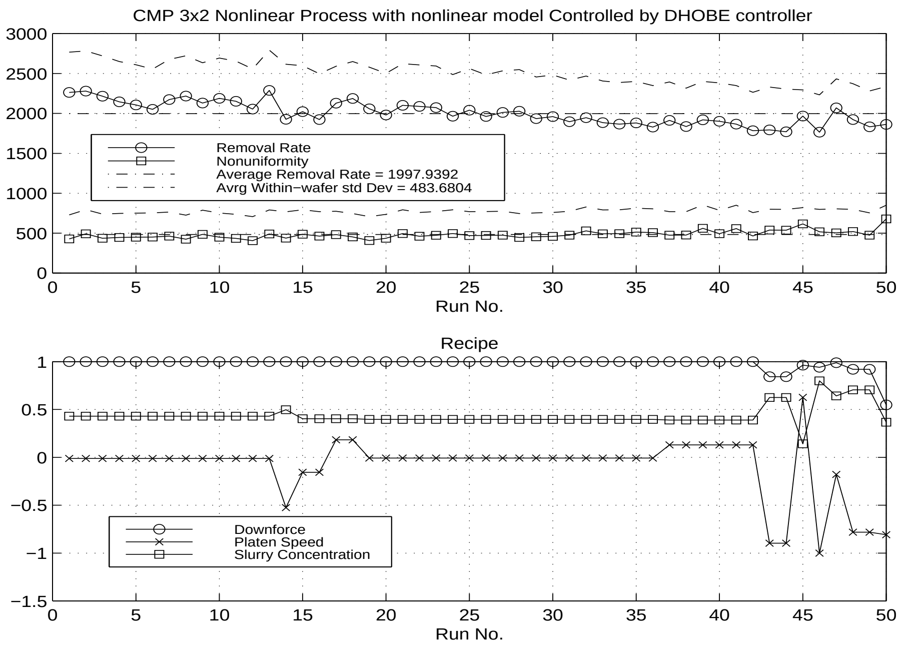

Figure 4.15: CMP3x2 Scenario 1 Controlled by DHOBE-MR

of adopting the nonlinear controller.

2) Quadratic Models (Scenario 1)

$$\begin{aligned} y_1 &= 2500 + 400u_1 + 500u_2 - 100u_3 - 800u_1^2 - 200u_2^2 + 1000u_3^2 - 40u_1u_2 - 100u_1u_3 \\ &\quad -350u_2u_3 - 7t \end{aligned}$$

$$\begin{aligned} y_2 &= 600 + 50u_1 + 100u_2 - 100u_3 - 200u_1^2 - 50u_2^2 + 300u_3^2 - 30u_1u_2 - 100u_1u_3 \\ &\quad -100u_2u_3 + 3t \end{aligned}$$

3) Linear Models (Scenario 2)

The following simulations used a linear model controller to compensate for the nonlinear process model. This also represents the case when using linear models only. The simulations were implemented using DHOBE-MR and SV controller

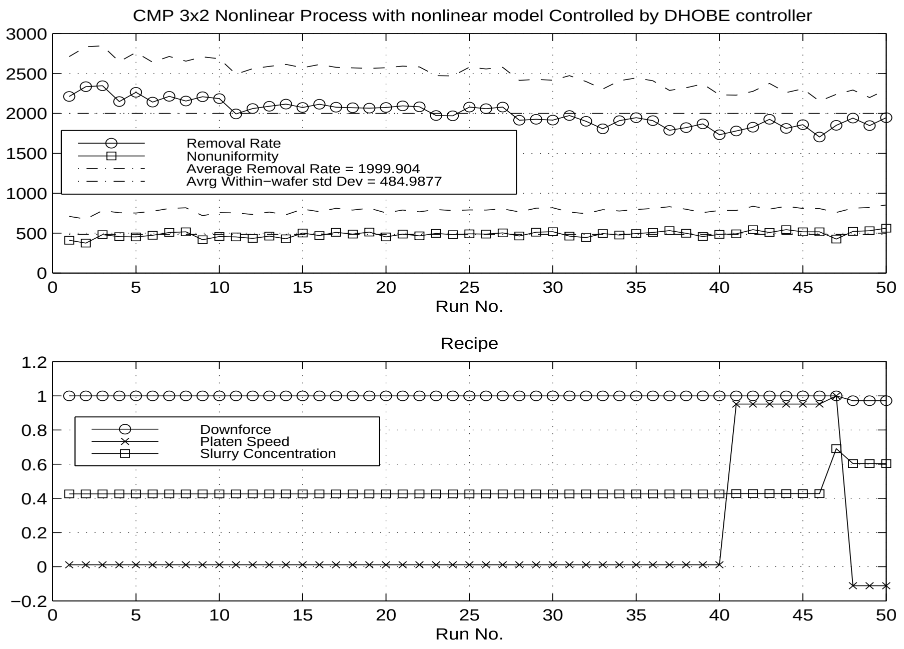

Figure 4.16: CMP3x2 Scenario 1 Controlled by DHOBE-SV

and the results are compared with OAQC method.

$y_1 = 2500 + 400u_1 + 500u_2 - 100u_3 - 7t$

$y_2 = 600 + 50u_1 + 100u_2 - 100u_3 + 3t$

From Table 4.6 it can be seen that when using the nonlinear model as the controller's model to compensate for the severe nonlinear processes, the mean value of response, standard deviation and mean square deviation are in the acceptable range. But when the internal model is linear, the compensation result is not good as expected. This also illustrates that the nonlinear internal model is necessary for such kind of cases no matter for which kind of control method.

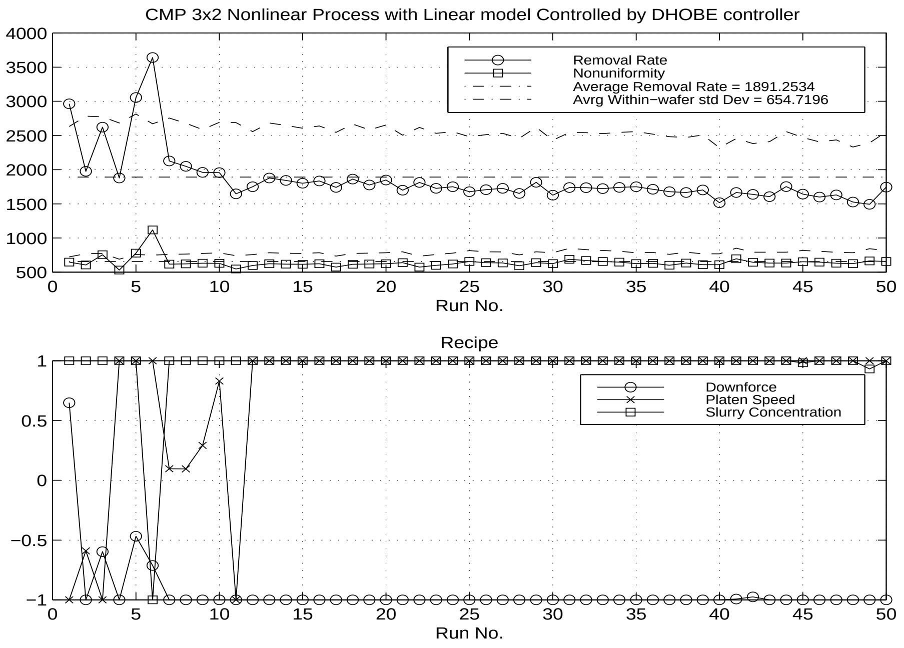

Figure 4.17: CMP3x2 Scenario 2 Controlled by DHOBE-MR

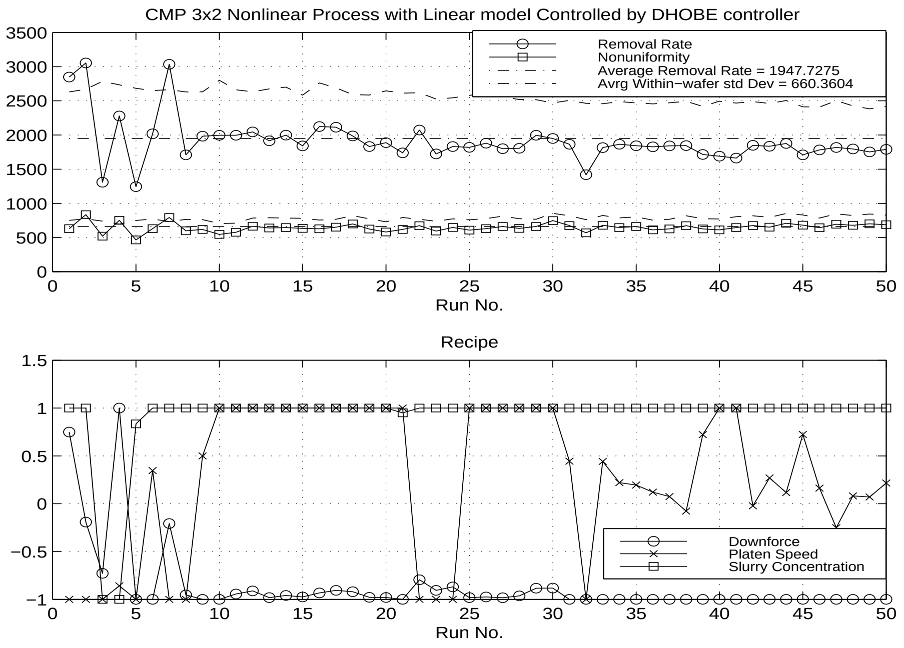

Figure 4.18: CMP3x2 Scenario 2 Controlled by DHOBE-SV

Scenario	Method	$\bar{y}_1$	$\bar{y}_2$	S_{y1}	S_{y2}	MSD_1	MSD_2
1	OAQC	2069.9	478.8	143.8	53.5	193.5	95.0
	DHOBE-MR	2005.5	490.5	139.7	41.2	235.9	98.6
	DHOBE-SV	2002.8	490.4	141.4	42.7	238.9	98.9
2	OAQC	1950.4	595.0	430.7	99.6	543.9	220.4
	DHOBE-MR	1921.5	663.9	457.0	99.9	568.4	271.9
	DHOBE-SV	1921.8	659.6	381.6	71.6	499.0	256.8

Table 4.6: OAQC and DHOBE Performance for CMP3x2 Models

Chapter 5

Conclusions and Future Work

We first analyzed the characteristic of semiconductor manufacturing processes. Models for these processes were further discussed in Chapter 1. Normally they can be described or approximated by the linear-in-the-parameter polynomial models. This gives us the direction and simplification of searching the model identification method. The normally used control method for semiconductor manufacturing is the run-by-run control. We discussed its features and explained several popular methods. Then we tried to apply the DHOBE method which is suitable for both model identification and for using the Set-Valued method with many advantages.

From the simulations in Chapter 4, we can see that the DHOBE controller can be applied to either linear or nonlinear process models. When comparing the simulation result with EWMA method, both methods achieved satisfactory result in linear models with slowly drifting, but DHOBE achieved better results in shorter transition phase caused by initial model error and faster tracking from large deviation caused by the step disturbance. The compensation for bad data case needs further improvement. The bad data can be considered as the outlier and there are suggested ways to screen out these bad data without taking them into the model identification process which will fool the controller into taking significant control actions.

The comparison using the same performance measures also shows that the controller is comparable to OAQC controller when it is applied to linear, almost linear or severe linear process models. When the process model is fully quadratic, the internal model within the controller is shown to be nonlinear necessarily in either cases.

The basic ideas for different model referenced RbR control methods are the same as illustrated in Chapter 2. Their difference is the method of updating the model at each run. The optimization step of each methods are almost the same. Then the model identification method will mainly decide the performance of the RbR controller. For the Set-valued method, it is different from the model reference method. As it applied the worst case approach, the control effect should be different from the other methods. However, from our simulations, it is almost the same as the model referenced method when using the same kind of model updating technology. This is because the DHOBE algorithm used the σ^2 to shrink the size of the feasible model parameter set, and this leads to the fact that the worst case parameter is almost the same as the center of the ellipsoid.

Bibliography

[1] S. Dasgupta and Y.F. Huang "Asymptotically Convergent Modified Recursive Least-Squares with Data-Dependent Updating and Forgetting Factor for Systems with Bounded Noise" *IEEE Transactions on Information Theory*, vol IT-33, no.3, pp383-392, May 1987.

[2] A.Rao and Y.F.Huang, "Tracking characteristics of an OBE parameter estimation algorithm." *IEEE Transactions on Signal Processing*,vol 41, pp. 1140-1148, Mar. 1993

[3] E.Fogel and Y.F.Huang, "On the Value of Information in System Identification - Bounded Noise Case." *Automatica*,vol 18, No.2, pp. 229-238, 1982

[4] S.G.McCarthy and R.B.Wells "Model Order Reduction for Optimal Bounding Ellipsoid Channel Models" *IEEE Transactions on Magnetics*, Vol 33 No.4 pp 2552-2568 Jul 1997.

[5] E.D.Castillo and J.Y.Yeh "An Adaptive Run-to-Run Optimizing Controller for Linear and Nonlinear Semiconductor Processes" *IEEE Transactions on Semiconductor Manufacturing*, Vol 11, No.2 pp285-294, May 1998.

[6] Z. Ning, J.R.Moyne, T.Smith, D.Boning, E.D Castillo, J.Y.Yeh and A. Hurwitz "A Comparative Analysis of Run-to-Run control Algorithms in the Semiconductor Manufacturing Industry." *7th Annual IEEE/SEMI Advanced Semiconductor manufacturing Conference*, pp375-381, Nov. 1996.

[7] J.Baras and N.S. Patel "Designing Response Surface Model-Based Run-by-Run Controller: A Worst Case Approach" *IEEE Transactions on Component, Packaging and Manufacturing Technology*, Part C, vol. 19, no 2, pp98-105, Apr. 1996.

[8] E. Fogel and Y.F. Huang "On the value of information in system identification-bounded noise case." *Automatica*, vol.18, no. 2 pp.229-238, Mar. 1982.

[9] T.H.Smith and D.S. Boning "Artificial neural network exponentially weighted moving average controller for semiconductor processes." *Journal of Vaccum Science Technology*, A 15(3) pp.1377-1384, May 1997.

[10] T.H.Smith and D.S. Boning "A self-tuning EWMA controller utilizing artificial neural network function approximation techniques." *IEEE/CPMT Int'l Electronics Manufacturing Technology Symposium*, pp.355-363, 1996.

[11] S. Leang and C.J. Spanos "Statistically Based Feedback Control of Photoresist Application." *IEEE/SEMI Advanced Semiconductor Manufacturing Conference*, pp.185-190, 1991.

[12] E. Sachs, A. Hu and A.Ingolfsson "Run by Run Process Control:Combining SPC and Feedback Control" *IEEE Transactions on Semiconductor Manufacturing*, Vol 8, No.1 pp26-43, Feb 1995.

[13] The Society of Chemical Engineers of Japan "Introduction to VLSI Process Engineering" *Chapman & Hall* 1993.

[14] D. Boning "Semiconductor Process Design: Representations, Tools, and Methodologies" Ph D. Thesis, MIT 1991

[15] D. Boning and et al "Run by Run Control of Chemical Mechanical Polishing" *IEEE/CMPT International Electronics Manufacturing Technology Symposium* , pp81-87, 1995.

[16] A. Hu and et al "Application of Run by Run Controller to the Chemical Mechanical Planarization Process" *IEEE/CHMT International Electronics Manufacturing Technology Symposium* , pp235-240, 1993.

[17] A. Ingolfsson and E. Sachs "Stability and Sensitivity of an EWMA Controller" *Journal of Quality Technology* , Vol. 25, No.4, pp271-287, Oct. 1993.

[18] J.S. Hunter "The Exponentially Weighted Moving Average" *Journal of Quality Technology* , Vol. 18, No.4, pp203-210, Oct. 1986.

[19] E.D. Castillo "A Multivariate Self-Tuning Controller for Run-to-Run Process Control under Shift and Trend Disturbances" *IIE Transactions*, pp1011-1021, Vol 28. 1996.

[20] M.F.Cheung, S. Yurkovich and K.M. Passino "An Optimal Volume Ellipsoid Algorithm for Parameter Set Estimation" *Proceedings of 30th IEEE Conference on Decision and Control*, pp969-974, 1991.

[21] E.D.Castillo and A.M.Hurwitz "Run-to-Run Process Control: Literature Review and Extensions" *Journal of Quality Technology*, Vol. 29, No. 2, pp184-196, April 1997.

[22] S.Adivikolanu and E. Zafiriou "Robust Run-to-Run Control for Semiconductor Manufacturing with Static Processes Models" *Technical Report*, 1997.