Complex 2004

Proceedings of the 7th Asia-Pacific
Conference on Complex Systems
Cairns Convention Centre, Cairns, Australia
6-10th December 2004

Implementing a Process Model for Real-Time Applications

Yu-Chu Tian[1], David C. Levy[2], and Tianlong Gu[3]
[1]*Centre for IT Innovation and School of Software Engineering and Data Communications, Queensland University of Technology, GPO Box 2434, Brisbane QLD 4001, Australia*
Email: y.tian@qut.edu.au
[2]*School of Electrical and Information Engineering, The University of Sydney, Sydney NSW 2006, Australia*
Email: d.levy@ee.usyd.edu.au
[3]*School of Computer Science, Guilin University of Electronic Technology, Guilin 541004, P.R. China*
Email: cctlgu@gliet.edu.cn

Abstract

Model-based advanced control of industrial processes requires well-developed process models. The established models are often complex with the features of multi-dimension, high nonlinearity, strong coupling, etc. Those features make the computational demand of the models conceptually high, especially for large-scale systems, resulting in difficulties in implementation of real-time control systems. This paper reports our efforts in overcoming the difficulties for a large-scale industrial process: a continuous hot dip galvanising production line. The process model is developed first, which is a set of partial differential equations. It is then transformed into a discrete state space model suitable for computation using digital computers. In order to meet the requirements of real-time control, algorithms are designed to significantly reduce the computational time. The process model and its computational algorithms are implemented and embedded into the real-time control system of the galvanising production line in the real plant.

1. Introduction

All systems must be logic correct. Real-time systems are time-critical systems in which timeliness is as important as the logic correctness. They are pervasive in the real world, e.g. in computing, communication, multimedia, robotics, process monitoring and control, etc. Due to the wide applications of the real-time systems and increasing demand on functionality and performance, real-time technology becomes an emerging and challenging area in computer engineering.

There have been some reports on real-time systems with regard to software architecture and software engineering, e.g. (Jeon and Choi, 2001; Carreira and Costa, 2003; Kumarakulasingam and Saiedian, 2002; Zalewski, 2001; Naedele, 2001). However, how to translate a real-time application problem into a form suitable for software specification, design, and implementation is still challenging. Moreover, reasonable simplifications of the original problem in this

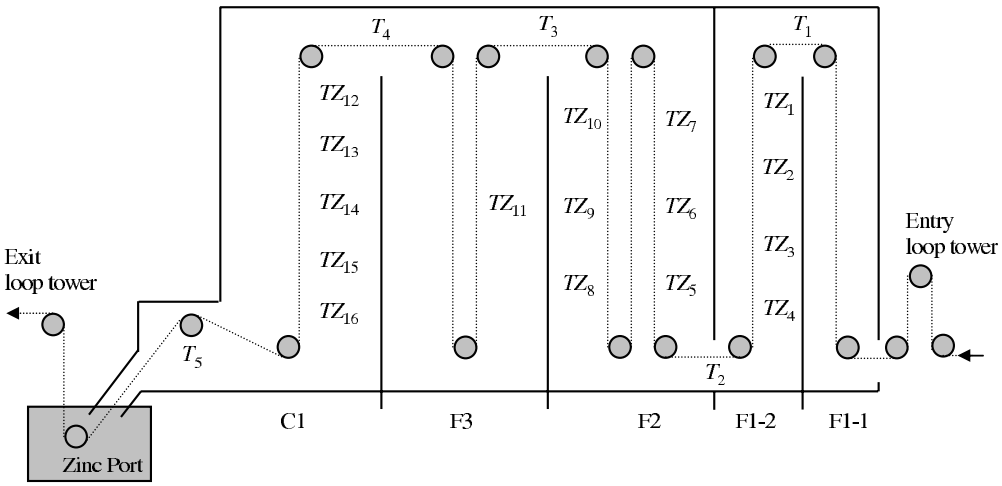

Figure 1. *Continuous annealing furnace in a galvanising process.*

translation process will lead to a significant reduction of computational demand and make the scheduling of various real-time tasks much easier. This work reports our efforts in this aspect through implementation of a mathematical model of a large-scale industrial process for real-time applications.

The investigated industrial process is a continuous hot dip galvanising production line in an iron and steel plant. A large-scale continuous annealing furnace, as shown in Figure 1, is the most important equipment in the production line. The furnace is about 42m high and 38m long. It consists of four sections: F1 (flame preheating), F2 (radiation heating), F3 (cooling), and C1 (gas-jet cooling).

A steel strip of the thickness between 0.3mm and 3mm passes continuously through the four consequent furnace sections for annealing and then through a zinc port for zinc coating. The equivalent length of the steel strip in the furnace is about 183m.

Maintaining steel strip temperatures T_1 through T_5 at specified values is critical for the quality of the galvanised steel products. The expected values of these temperatures are determined by the so-called annealing diagram, which specifies a set of strip temperatures in the furnace for each of the steel strip rolls. Therefore, many efforts have been made to control the steel strip temperature distribution in furnace, e.g. (Yoshitani and Hasegawa, 1998).

The strip temperatures in furnace are controlled through manipulating furnace temperatures TZ_1 through TZ_{16}, while the furnace temperatures are regulated through combustion control. Therefore, the real-time control system of the furnace has a hieratical and cascade structure. For steel annealing diagram no. 76, a typical furnace temperature profile is depicted in Figure 2. Typical strip and furnace temperature profiles at the temperature measurement points in the furnace are tabulated in Table 1. More information about the production line can be found in (Tian et al., 2000; Tian and Tadé, 2002).

For advanced control of the strip temperature in furnace, a mathematical model of the process is required. The model is used to predict the strip temperature distribution in furnace several steps ahead. The prediction is then used for control design. It has been clarified from system analysis that a good choice of the control period for the system is 4s, implying that the model computation time must be much shorter than the control period, i.e. 4s.

However, a mathematical model for such a large-scale galvanising process is normally a set of partial differential equations, which are computationally intensive. Using conventional

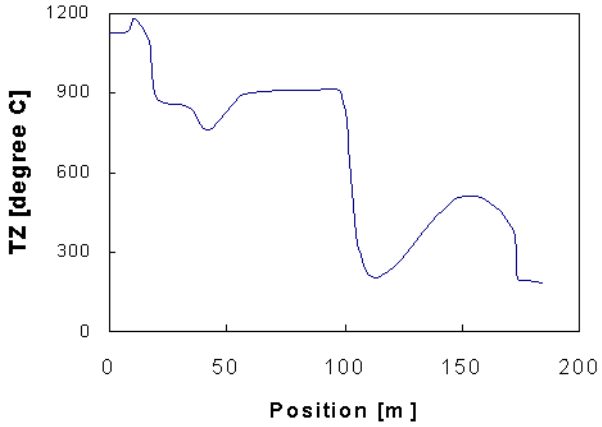

Figure 2. *A typical furnace temperature profile.*

Table 1. *Typical strip and furnace temperature profiles at measurement points (unit: °C).*

Strip temperatures $T_1 \sim T_5$	450	650	760	630	465	
F1 furnace temperatures $TZ_1 \sim TZ_4$	1127	1131	1181	1097		
F2 furnace temperatures $TZ_5 \sim TZ_{10}$	882	847	764	895	912	888
F3 furnace temperature TZ_{11}	210					
C1 furnace temperatures $TZ_{12} \sim TZ_{16}$	507	413	205	195	182	

numerical methods, it is difficult to complete the model computation in a control period (4s) with the required computation accuracy. Consequently, feasible scheduling strategies are not available for such a real-time computing application. Therefore, the mathematical model has to be reasonably and significantly simplified and efficient algorithms have to be developed for model implementation.

System analysis and the main aspects of the model development for the galvanising process have been discussed in (Tian et al., 2000; Tian and Tadé, 2002). As a continuation of our previous work in this area, this paper focuses on the implementation aspect of the developed process models for real-time applications through further model simplification, and algorithm development.

To make this paper self-contained, let us have a brief discussion of the model development of the galvanising process in the next section.

2. Development of Process Models

2.1 Partial Differential Equation Model

To describe the strip temperature distribution T in the furnace, a fixed two-dimensional Cartesian coordinate system is taken with x and y representing the thickness and length directions, respectively, as shown in Figure 3. The physical properties of the steel strip are described by its specific heat C [J/kgK], density ρ [kg/m^3], and heat conductivity K_s [W/mK].

Under some reasonable assumptions, the strip temperature distribution T can be described

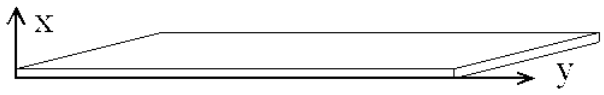

Figure 3. *Fixed two-dimensional coordinate system.*

by a partial differential equation (PDE) (Tian et al., 2000)

$$\frac{\partial T(x,y,t)}{\partial t} = \frac{K_s}{C\rho}\left\{\frac{\partial^2 T(x,y,t)}{\partial x^2} + \frac{\partial^2 T(x,y,t)}{\partial y^2}\right\} \tag{1}$$

with the initial condition

$$T(x,y,0) = T0(x,y) \tag{2}$$

and "linearised" boundary conditions

$$\begin{cases} \dfrac{K_s}{h(y,t)}\dfrac{\partial T(x,y,t)}{\partial x}\Big|_{x=0} = T(0,y,t) + T_Z(y,t) \\ \dfrac{K_s}{h(y,t)}\dfrac{\partial T(x,y,t)}{\partial x}\Big|_{x=d} = T_Z(y,t) - T(d,y,t) \end{cases} \tag{3}$$

where T_Z represents furnace temperature, h [W/m^{2}K] is an equivalent heat transfer coefficient between furnace and strip, $x \in [0,d]$ (d is the strip width), and $y \in [0,L]$ (L is the equivalent length of the strip in furnace). Some simplifications have been made in deriving this PDE model in order to implement the model in real-time (Tian et al., 2000).

The average strip temperature along the strip thickness direction is obtained through

$$T_S(y,t) = \frac{1}{d}\int_0^d T(x,y,t)dx \tag{4}$$

2.2 State Space Model

The above PDE model equations (1) through (4) developed from the first principles are difficult to solve analytically, necessitating numerical computation. Therefore, the PDE model needs to be transformed into a form suitable for real-time numerical computation.

Among various numerical computation methods, the finite difference technique is adopted in this work to develop a simple yet practical state space model for strip temperature distribution. The model is discrete in both time and space. The main steps of the model development are stated below.

Firstly, partition x into N_x sections of length Δx and y into N_y sections of length Δy, respectively, where $N_x \times \Delta x = d$ and $N_y \times \Delta y = L$. This results in $N_x \times N_y$ parallelograms on the xy space, as shown in Figure 4. For simplicity, let i, j denote $i\Delta x$ and Δy, respectively, without confusion.

Secondly, discretise the time t into a finite number of instants with the time step of Δt. For simplicity, let k denote $k\Delta t$.

Then, use forward and backward differences to approximate $\dfrac{\partial T(x,y,t)}{\partial t}$ and $\dfrac{\partial T(x,y,t)}{\partial y}$, respectively. However, use forward-backward differences to approximate $\dfrac{\partial T^2(x,y,t)}{\partial x^2}$ and $\dfrac{\partial T^2(x,y,t)}{\partial y^2}$.

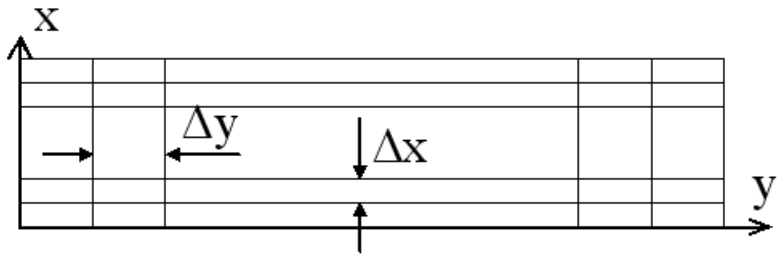

Figure 4. *Space partition.*

After that, add up the finite number of strip temperatures at the discretised x points to approximate the integration $\int_0^d T(x,y,t)dx$.

Substituting those approximations of the partial derivatives and integration into the original PDE model equations (1) through (4) gives a set of coupled difference equations. With some algebraic operations, the following state space model can be derived, which is discrete in both time and space (Tian et al., 2000)

$$X(k+1) = A(k)X(k) + B(k)U(k) \tag{5}$$

where

$$\begin{cases} X(k) &= [T_S(0,k), \cdots, T_S(N_y,k)]^T \\ U(k) &= [T_Z(0,k), \cdots, T_Z(N_y,k)]^T \\ B(k) &= \text{diag}(d_{0k}, \cdots, d_{N_y k}) \\ A(k) &= \begin{bmatrix} a_{11} & b & & & & & \\ b+c & a_{22} & b & & & & \\ & b+c & a_{33} & b & & & \\ & & \cdots & \cdots & \cdots & & \\ & & & & b+c & a_{N_y,N_y} & b \\ & & & & & b+c & a_{N_{y+1},N_{y+1}} \end{bmatrix} \end{cases} \tag{6}$$

d_{jk} is expressed by

$$d_{jk} = \frac{2}{C\rho d} h(j,k)\Delta t, j \in \{0,1,\cdots,N_y\} \tag{7}$$

The matrix $A(k) = [a_{pq}]$ is a N_{y+1} by N_{y+1} square matrix with most elements being zero

$$\begin{cases} a_{pp} = 1-b-d_{0k} & p=1 \\ a_{pp} = 1-2b-c-d_{(p-1)k}, & p \in \{2,\cdots,N_y\} \\ a_{pp} = 1-b-c-d_{N_y k}, & p = N_y+1 \\ a_{p(p+1)} = b, & p \in \{1,\cdots,N_y\} \\ a_{p(p-1)} = b+c, & p \in \{2,...,N_y+1\} \\ a_{pq} = 0, & \text{otherwise} \end{cases} \tag{8}$$

where

$$b = \frac{K_s}{C\rho}\frac{\Delta t}{(\Delta y)^2} \tag{9}$$

$$c = v(k)\frac{\Delta t}{\Delta y} \tag{10}$$

The variable $v(k)$ represents the strip speed in m/s. It is an operational variable in production but not a manipulated variable in control design and implementation of the system.

2.3 Settings of Model Parameters

The developed state space model expressed by equations (5) through (10) contains some parameters representing the physical properties and operating conditions, such as C, ρ, d, v, etc. These parameters are not adjustable in model computation unless the operating environment has changed. In addition to these parameters, there are two adjustable parameters, Δt and Δy, which determine the computational complexity, accuracy, and stability of the model. Big values of Δt and Δy reduce the computational demand while resulting in less accurate results and probably unstable model computation. On the other hand, small values of Δt and Δy benefit the computation accuracy and stability; however, this will lead to a significant increase in computational demand and consequently the real-time requirement may not be met. Therefore, a compromise has to be made when selecting Δt and Δy values.

A detailed analysis of the computational stability of the state space model has been carried out. The stability conditions are listed below (Tian et al., 2000)

$$\begin{cases} h(j,k)\Delta t < C\rho d \ \text{ for} j \in [1, N_y - 1] \\ 2h(N_y,k)\Delta t < \left[2 - v\dfrac{\Delta t}{\Delta y}\right] C\rho d \\ v\frac{\Delta t}{\Delta y} C\rho d < 2h(0,k)\Delta t < \left[2 + v\frac{\Delta t}{\Delta y}\right] C\rho d \end{cases} \tag{11}$$

From the stability conditions in equation (11), the full range of operating conditions and physical properties of the steel strip has been investigated for determination of reasonable values of Δt and Δy. In the final implementation of the state space model, it has been chosen that Δt = 4s and Δy = 10m.

3. Algorithm Design and Implementation

3.1 Standard Least Square Algorithms

When using the developed state space model to predict the strip temperature distribution and further for control design, all model parameters in the matrices A and B must be known. However, the operating conditions of the system keep changing continuously. Therefore, a set of fixed values for model parameters is not acceptable for online and real-time applications. Some model parameters must be identified online and in real-time.

The fundamental algorithms for model identification are simple. For example, the standard least-square identification algorithm and its extensions can be used. For easier discussion of the algorithm developed in this work, a brief description of the standard least square identification algorithms is given below.

Consider a model

$$z(t) = \phi_1(t)\theta_1 + \phi_2(t)\theta_2 + \cdots + \phi_n(t)\theta_n = \phi^T(t)\theta \tag{12}$$

where ϕ and θ are regressor and parameter vectors, respectively. The model is linear in the parameters.

A series of observations of z is obtained together with corresponding ϕ, $\{(z_i, \phi_i), i = 1, 2, \cdots, t\}$.

Denote

$$\begin{aligned} Z(t) &= [z(1), z(2), \cdots, z(t)]^T \\ E(t) &= [\epsilon(1), \epsilon(2), \cdots, \epsilon(t)]^T \\ \epsilon(i) &= z(i) - \hat{z}(i) = z(i) - \phi(i)\theta; i = 1, ..., t \end{aligned} \tag{13}$$

Minimising the performance index $\sum_i \epsilon(i)$ with respect to the parameter vector θ gives the estimate of the parameters

$$\begin{cases} \hat{\theta}(t) = P(t)\Phi^T Z = P(t)\sum_i \phi(i)z(i) \\ P(t) = \left(\Phi^T\Phi\right)^{-1} = \left(\sum_i \phi(i)\phi^T(i)\right)^{-1} \\ \Phi = \left[\phi^T(1), \phi^T(2)\cdots, \phi^T(t)\right]^T \end{cases} \tag{14}$$

The solution of $\hat{\theta}$ in (14) requires that $\Phi^T\Phi$ is non-sigular.

To recursive form of the above least square estimation of θ is expressed by

$$\begin{cases} \hat{\theta}(t) = \hat{\theta}(t-1) + K(t)\left[z(t) - \phi^T(t)\hat{\theta}(i-1)\right] \\ K(t) = P(t)\phi(t) = \dfrac{P(t-1)\phi(t)}{1 + \phi^T(t)P(t-1)\phi(t)} \\ P(t) = \left[I - K(t)\phi^T(t)\right]P(t-1) \end{cases} \tag{15}$$

To weight more new data and forget old history, a forgetting factor λ is introduced into the estimation scheme. The corresponding estimation $\hat{\theta}$ is

$$\begin{cases} \hat{\theta}(t) = \hat{\theta}(t-1) + K(t)\left[z(t) - \phi^T(t)\hat{\theta}(i-1)\right] \\ K(t) = P(t)\phi(t) = \dfrac{P(t-1)\phi(t)}{\lambda + \phi^T(t)P(t-1)\phi(t)} \\ P(t) = \left[I - K(t)\phi^T(t)\right]P(t-1)/\lambda \end{cases} \tag{16}$$

where the forgetting factor $\lambda \in [0, 1]$, usually $\lambda \in [0.95, 1]$.

For the developed state space model (5) through (10), recall that Δy is determined to be 10m, implying that A is a 19 by 19 matrix. Because the above-mentioned standard least square algorithms are applied to each of the 19 rows of the matrix A, there will be a demand of heavy computation of matrices and matrix inverses in estimation of A. This prevents us from using the standard algorithms for real-time implementation of the state space model, necessitating more efficient algorithms.

3.2 Development of a More Efficient Algorithm

It is seen from equation (8) that the matrix A is sparse with most of its elements being zero. The first and last row of A has respective 2 elements, and all other rows have only 3 elements! This structural feature of A can be used to develop some algorithms with significant reduction of computational demand.

The following is a brief description of such an algorithm developed in this work. The basic framework of the algorithm is similar to (16), i.e. the least square scheme with a forgetting factor. However, the calculation for each row of A is carried out over 2 or 3 elements, compared to 19 elements of the standard algorithms! In the following descriptions of the developed algorithm, the vector θ is formed by the 2 or 3 non-zero elements of a row of A while $\phi(t)$ is formed by the corresponding 2 or 3 elements of the state vector $X(t)$. The measurement $z(t)$, i.e. T in the model of the steel strip temperature distribution, corresponds to a specific point of y.

Step 1. For each row of A, construct ϕ and θ using the 2 or 3 non-zero elements in the corresponding row of the matrix A.

Step 2. For each row of A, calculate $K(t)$ based on $\phi(t)$ and $P(t-1)$. The formula is

$$K(t) = \frac{P(t-1)\phi(t)}{0.95 + \phi^T(t)P(t-1)\phi(t)} \tag{17}$$

Step 3. For each row of A, update P based on $K(t)$ and ϕ. The equation is

$$P(t) = \left[I - K(t)\phi^T(t)\right] P(t-1)/0.95 \tag{18}$$

Step 4. For each row of A, identify the 2 or 3 non-zero elements that form the corresponding θ. The formula is

$$\hat{\theta}(t) = \hat{\theta}(t-1) + K(t)\left[z(t) - \phi^T(t)\hat{\theta}(i-1)\right] \tag{19}$$

These steps form the fundamental algorithm for model identification in this work.

3.3 Further Simplification

The developed model identification algorithm can be further simplified. It is known from the system analysis that not all non-zero elements of the matrix A need to be identified!

Let us consider the row $r(1 < r < 19)$ of the matrix A. The 3 non-zero elements in the row are

$$\begin{cases} a_{(r-1)r} = b + c, \\ a_{rr} = 1 - 2b - c - d_{(r-1)k}, \\ a_{r(r+1)} = b \end{cases} \tag{20}$$

From equation (10), $c = v\Delta t/\Delta y$ can be calculated directly because the strip speed v is exactly known and Δt = 4s and Δy = 10m are predetermined model parameters. It is seen from equation (20) that $a_{(r-1)r} = a_{r(r+1)} + c$. This means that either $a_{(r-1)r}$ or $a_{r(r+1)}$ can be excluded from the identification scheme as they can be derived from each other with the known value of c. Therefore, only 2 elements need to be identified for each row of the matrix A! The operations of 2 by 2 matrices are very easy.

In summary, with the developed system identification algorithm, the operations of 19-by-19 matrices are simplified into the operations of 19 2-by-2 matrices for the state space model (5) through (10). This is the best result that we have obtained so far for this application.

3.4 Implementation of the Identification Scheme

In programming, the calculation of various vectors has been transformed into scalar calculations within structured loops; and for each row of the matrix A, only 2 or 3 non-zero elements are taken into account.

For equation (19), the psudo-code looks like

```
For row r (1 < r <19)
  for (col=r-1, col<=r+1, col++)
    a[r,col]+=K[col]*(T-Tpre);
```

Other parts of the identification algorithm can also be programmed easily.

4. Prediction of Strip Temperature Distribution

With the developed state space model (5) through (10) and the identified model parameters, the prediction of the steel strip temperature distribution in furnace is straightforward. Although the calculation can be carried out based on the 19-by-19 matrix A, much less computation can be expected because there are only 2 or 3 non-zero elements in each row of A. This can be easily implemented in programming.

Let us consider the first row of the matrix A. The prediction is made using the following simplified formula

$$x_1(k) = a_{11}x_1(k-1) + a_{12}x_2(k-1) \tag{21}$$

For rows 2 through 18 of the matrix A, the computation is carried out using a three-term formula

$$x_r(k) = \left[a_{(r-1)r}, a_{rr}, a_{r(r+1)}\right] \begin{bmatrix} x_{r-1}(k-1) \\ x_r(k-1) \\ x_{r(r+1)}(k-1) \end{bmatrix} \tag{22}$$

For the last row of the matrix A, the prediction is expressed by

$$x_{19}(k) = a_{18}x_{18}(k-1) + a_{19}x_{19}(k-1) \tag{23}$$

5. Concluding Remarks

As mentioned earlier, the control period of the overall control system for the galvanising process is 4s, which is determined from system analysis and must be met in control system design and implementation. Numerical computation of the original partial differential equation model of the process requires a much longer time than the specified control period, making it impossible to schedule the control tasks.

Efforts have been made in this work to significantly reduce the computational demand through model simplification and implementation. The proposed computational scheme is the operations of 19 2-by-2 matrices. The developed models, the algorithms for parameter identification and strip temperature prediction, and the implementation of the models and algorithms are verified through simulation and industrial applications. An evaluation of the timeliness of the proposed approach shows that the overall model computation, including system identification and prediction, can be completed within 400ms on the control computer in the real plant. This satisfies the requirement of the real-time control, and makes the control tasks of the overall control system schedulable.

The developed models and algorithms for model computation are successfully embedded into the real-time control system of the galvanising production line. Industrial applications as well as simulation studies reveal that the implemented process model has the accuracy suitable for real-time control. For strip temperatures T_2 through T_5, the means of the predictive errors over three hours are 0.03, -0.03, 0.02, and 0.05, respectively. The corresponding standard deviations are 1.59, 0.36, 0.38, and 0.68, respectively. A plot of the redictive error for T_2 over 3 hours is shown in Figure 5.

Through this work, it is found that when a process model is descretised using a properly designed method, a state space model discrete in both time and space can be obtained with a special structure in the state matrix: the matrix is sparse with most elements being zero. Taking into account this structural feature of the state matrix, it is possible to develop efficient

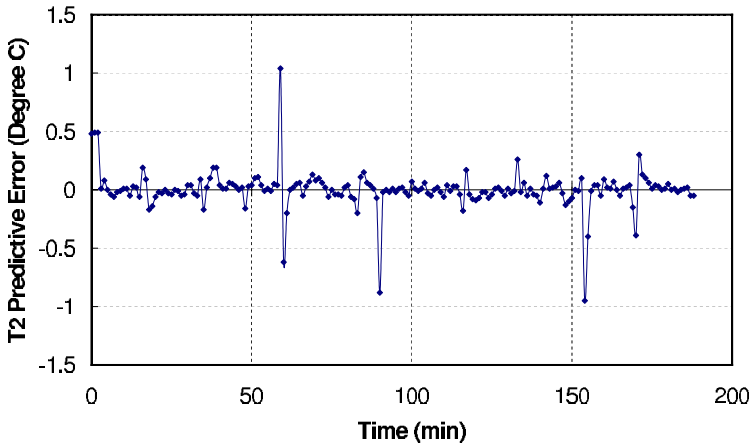

Figure 5. *Predictive errors of T_2 over 3 hours.*

algorithms for model identification and model-based prediction. This work has shown how to develop such algorithms and how to implement these algorithms.

Acknowledgments

The first author (YCT) would like to thank CH Hou and XQ Xie, who were respectively working towards their Master and Bachelor degrees in this project, for their assistance in development, verification, and implementation of the process models.

References

Carreira, P. J. F. and Costa, M. E. F. (2003). Automatically verifying an object-oriented specification of the steam-boiler system. *Sci of Computer Programming*, 46:197–217.

Jeon, Y. and Choi, C.-H. (2001). Minimization of blocking time in component-based software architecture for control systems. *Control Eng Practice*, 9:683–693.

Kumarakulasingam, P. and Saiedian, H. (2002). A framework for evaluating the effectiveness of real-time object-oriented models. *Infor & Software Tech*, 44:395–404.

Naedele, M. (2001). An approach to modeling and evaluation of functional and timing specifications of real-time systems. *J of syst & Software*, 57:155–174.

Tian, Y.-C., Hou, C. H., and Gao, F. (2000). Mathematical modelling of a continuous galvanising annealing furnace. *Dev Chem Eng Mineral Processing*, 8(1):359–374.

Tian, Y.-C. and Tadé, M. O. (2002). Processing systems engineering: successful application in an industrial process. *Dev Chem Eng Mineral Processing*, 10(1):181–196.

Yoshitani, N. and Hasegawa, A. (1998). Model-based control of strip temperature for the heating furnace in continuous annealing. *IEEE Trans Control Syst Tech*, 6(2):146–156.

Zalewski, J. (2001). Real-time software architectures and design patterns: fundamental concepts and their consequences. *Annual Rev in Control*, 25:133–146.