1 Injection Molding: Background

1.1 Plastic Materials and Properties

Human history has been defined in terms of materials categories: the Stone Age, the Bronze Age, and the Iron Age. It is well accepted that we are now living in a polymer age. Since the 20th century, polymer materials, including plastics, fibers, elastomers, and proteins, have gradually appeared in almost every area of people's everyday life, and there are a variety of applications in agriculture, industry, and even the defense industry. In all of the polymer materials, plastic is a major class.

Plastics are ubiquitous in modern society, with applications ranging from toys to electronic components, interior or structural parts of automobiles, and different components in trains and airplanes. There is hardly an area that does not use plastic parts in modern industry. The main advantages of plastic materials compared to other commonly used materials such as metal and woods are obvious. First of all, they have good physical or chemical properties, such as low density (light weight), chemical resistance, and durability, and are thermostatically and electrically insulating. Second, they are economical in producing massive quantities of products. Third, plastic materials are normally easy to fabricate, especially compared to metal, and the energy cost accompanying plastics processing is also significantly reduced. Although plastics also have some disadvantages, such as not being biodegradable and promoting crude oil mining, these problems could be solved with recycling and the development of biodegradable plastics and other environmentally friendly enhancements.

The applications for plastics in modern industry and in people's everyday life are almost limitless. Plastic products can be found everywhere. The largest application of plastics worldwide is the packaging industry, including numerous products like containers, bottles, drums, trays, boxes, cups and vending packaging, baby products, and protection packaging. The typical materials used in this area are low-density polyethylene, high-density polyethylene, polypropylene, polystyrene, and polyethylene terephthalate.

The second largest consumer of plastic products is the building and construction industry. Since plastic materials have great versatility and a combination of excellent strength-to-weight ratio, durability, cost effectiveness, corrosion resistance, and low maintenance cost, they are an attractive choice throughout the construction sector. Pipes, siding sheets, insulators, roofing materials, and door and window frames all use plastics, and the materials used are not limited to only those used in the packing industry, but may also be some other types, like polyvinyl chlorides.

Another important application area is electrical and electronic devices. The wires of most electronic devices are encased in plastic. Most outer casings of electronic devices, such as telephones, mobile phones, lighting fixtures, fans, computers, and televisions, are plastic as well. Plastic materials like polycarbonate, polyamide, acrylonitrile butadiene styrene, and styrene acrylonitrile are widely used in this area.

A variety of plastics have been used in the manufacture of automobiles, trucks, trains for high-speed railways, and even airplanes. Nowadays many critical parts of the automobile are made of plastic materials, such as the steering wheels, air-intake manifolds, windshield wipers, bumpers, headlights, fuel pumps, and emission canisters. The train windows of China Railway High-speed (CRH) are made of six layers, where two of them are plastic materials [1]. This kind of multilayer structure can also be found in the aircraft windshield, for example, as illustrated in Fig. 1.1.

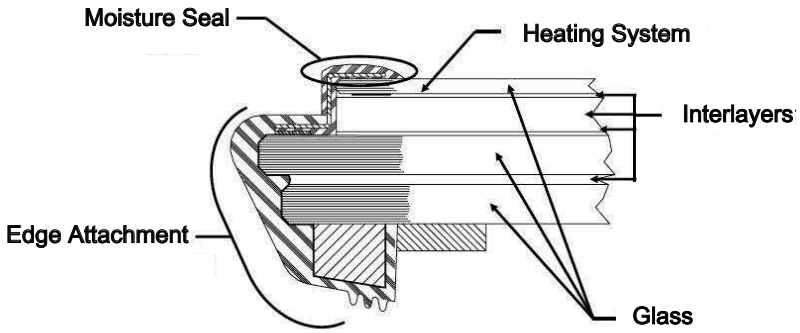

Figure 1.1 Structure of Airbus SA windshield [2]

The transparent interlayers in this schematic are plastic materials used to bond the glass plies and other components together to form the aircraft windows. They also contribute to enhancing the windows' pressure fail-safe capability and bird-impact resistance.

Owing to the rapid development of plastic materials and their applications in recent decades, this material family is still growing. In recent years, materials scientists and engineers developed the shape-memory plastic materials, which are polymeric smart materials that have the ability to return from a deformed state induced by an

external stimulus, such as temperature change [3], to their original or permanent shape. With the boom of nanotechnology, nanoplastic becomes a new engineering area. With the nanostructure units of nanocomposites added to the original plastic materials, the resulting nanoplastic can have some unique features, such as antibacterial, water resistance, high temperature tolerance, and high strength [4].

Plastic is a kind of organic polymer material with giant molecules, whose basic components are hundreds or thousands of monomers prepared synthetically or semisynthetically, mostly organic molecules such as carbon, hydrogen, oxygen, or nitrogen. Since plastics is a material group with vast variety, it can also include inorganic constituents, such as sulfur, chlorine, fluorine, or bromine.

Using polyethylene (PE) as an example, Fig. 1.2(a) shows the structure of PE. This structure can be shown in a more concise form as Fig. 1.2(b). It is clear that the molecular chain is constructed of many small molecules of C_2H_2, i.e., ethylene monomers. The notation "*n*" in Fig. 1.2(b) represents the number of repetitive units in the PE structure; its value could range from a few hundred to several thousand. Therefore, the molecular weight of PE may also be an undetermined value with a statistical distribution [5].

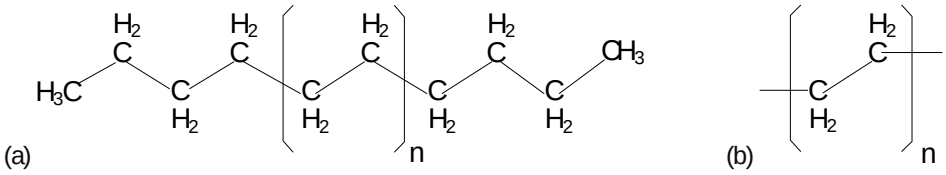

Figure 1.2 Molecular structure of polyethylene (PE)

1.1.1 Plastics Classification

Plastics may be classified in many ways, based on the molecular structure, processability, method of synthesis, and so on.

1.1.1.1 Molecular Structure

In terms of the molecular structure, plastics can be roughly divided into linear, branched, and cross-linked structures. A schematic illustration of these typical structures can be found in Fig. 1.3, where the lines represent the polymer molecules, and the black dots indicate the monomers [6].

A linear structure [7] has repeating units linked end-to-end together in a continuous length. Each monomer in the linear plastic is thus linked to only two others, resulting in a long and narrow molecule. Poly(vinyl chloride) (PVC), high-density polyethylene (HDPE), and some polyamides (nylon) are typical linear plastics.

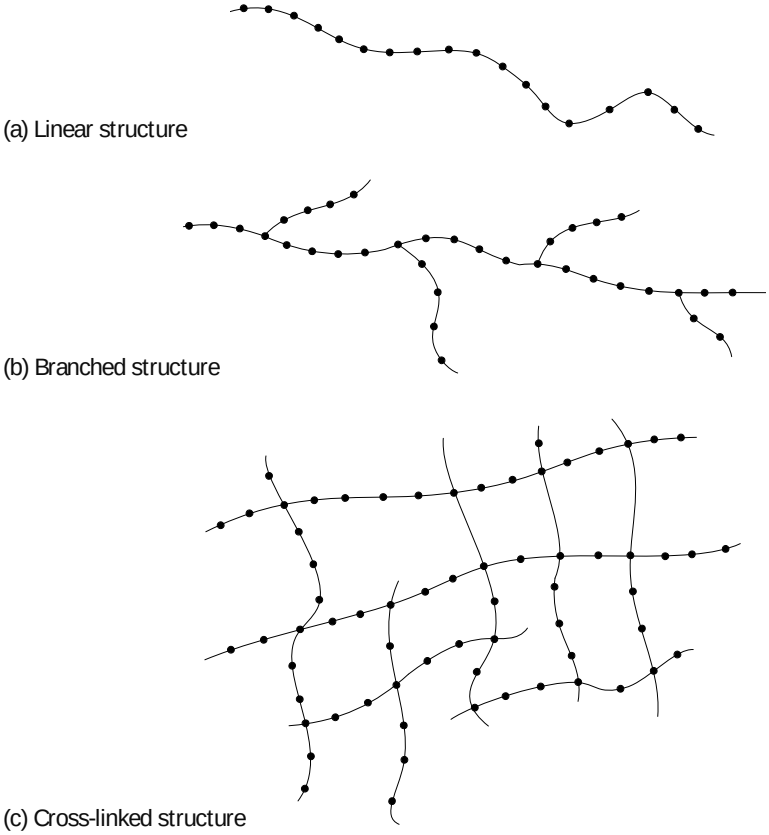

Figure 1.3 Different structures of plastic materials

A branched structure refers to those plastic materials with branches distributed irregularly in the backbone molecular chain. These branches make it difficult to pack them in an array, so they are less dense and less crystalline and easier to process. The amount and type of the branches could affect some important physical properties of the material, such as viscosity and elasticity. Low-density polyethylene (LDPE) is a typical branched material. Although both HDPE and LDPE have ethylene as the basic unit, their structural difference leads to significantly different physical properties and applications.

A cross-link is a bond that links one polymer chain to another. The cross-linked structure contains short side chains or cross-links that connect different molecule chains into a network-fashion microstructure. The cross-linking structure makes the plastic material more elastic, meaning that when a force is exerted on the cross-linked material it could deform and then return to its original state after the force is removed. Polyacrylamide and cross-linked polyethylene are typical cross-linked plastics [5].

1.1.1.2 Processability

Another classification based on processability divides the plastic materials into thermoplastic and thermoset. Briefly, the linear, branched, and slightly cross-linked plastic materials form the thermoplastics, while material that is heavily cross-linked after shaping of the products is known as a thermoset.

A thermoplastic is a plastic material that is solid and possesses significant elasticity at low temperature and can turn into a viscous liquid-like state at a high temperature, and this change is reversible. Since the thermoplastics can be softened and made to flow with the application of heat, they can be fabricated by processes like injection molding, extrusion, and blow molding. The molecular chains of a thermoplastic material are associated through intermolecular forces, hence allowing the thermoplastic to be remolded because the association force increases upon cooling and restores the bulk properties. This class of plastics includes PE, polypropylene (PP), polystyrene (PS), polycarbonate (PC), and acrylonitrile butadiene styrene (ABS).

Thermoset resin is a class of petrochemical in a soft solid or viscous state that changes irreversibly into an infusible, insoluble polymer network by curing. Curing can be induced by the action of heat or suitable radiation, or both. A cured thermosetting resin is called a thermoset. Prior to curing, the thermoset material is generally liquid or malleable with only partially polymerized states. The cure is actually a cross-linking process to form a network material. Generally, thermoset materials are stronger than thermoplastic materials because the molecular chain of the cured thermosets are connected by a three-dimensional network of bonds. However, the thermosets are more brittle, and their shapes are permanent, so they cannot be recycled to make new products. The fabrication of thermoset material must be done prior to the curing through, for example, reactive injection molding, transfer molding, extrusion, compression molding, or spin casting. Typical thermoset materials are polyurethanes (PU), polyimide (PI), and polyester resins.

1.1.1.3 Method of Synthesis

All plastics are synthetic or manufactured materials, and polymerization is the most important process of converting monomer molecules through a chemical reaction into polymer chains or three-dimensional networks. Depending on the method of polymerization, plastics can also be classified into addition plastics and condensation plastics.

Addition plastics are formed by additional polymerizations or reactions, where many monomers bond together via rearrangement of bonds without the loss of any atom or molecule. For example, the following polymerization (Fig. 1.4) of polyethylene is an addition reaction [8]:

2 Feedback Control Algorithms Developed for Continuous Processes

2.1 Introduction of Feedback Control Background

Control of the injection molding process not only involves sequence manipulation, but also some key process variable regulation. The following discussion will mainly focus on the process variable control. Early injection molding controllers were mainly constructed using simple electrical components such as timing relays and switches. The control is thus naturally open-loop. As the molding process becomes more and more complex, manufacturers require greater accuracy and tighter tolerances. All of the variations and disturbances during the molding process must be properly dealt with, so closed-loop control becomes necessary, which also triggers the application of computer-based controllers to the injection molding machines. It is therefore essential to give a brief introduction to some basic control concepts.

A controlled process is the object or process of the control; if a certain variable is the control goal, it can also be referred to as a controlled variable. The controlled process or controlled variable is the major concern of the controller. The controlling variable or control variable is the one that can be changed or entered from outside that manipulates the controlled process or variable. The value of the control variable is actually the control decisions to be made by the controller. Disturbances are defined as those variables aside from the control variable that cause the controlled process or variable to deviate from the set point. Using the injection molding machine barrel temperature control as an example, the schematic of a control loop is illustrated in Fig. 2.1. In this case, the controlled variables are the different zones' barrel temperatures, the control variables are the powers for different heaters, and disturbances are the temperature of the environment, the air flow in the factory, and others.

The control system can be categorized as manual control and automatic control. Some slow processes can be adjusted manually by human operators, and this kind of control is known as manual control. In injection molding, the manual control mode is still useful in sequence control during machine setting or problem solving.

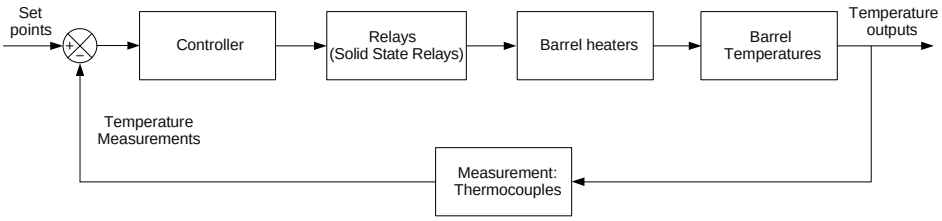

Figure 2.1 Illustration of injection molding barrel temperature control

During normal production, since modern injection moldings are fast processes, some even with a total cycle time as short as several seconds, it is extremely difficult for the operator to maintain the machine sequence and perform key variable control. The machine and process must be controlled automatically with a mechanical or electrical controller. The objective of the automatic controller is to manipulate the controlled variable to maintain the controlled variable at its set point in spite of various disturbances.

There are some basic elements for the automatic control system to function properly: a measurement unit, a controller, and a control actuator. The measurement unit often includes a sensor that converts the controlled variable to an electrical or some other measurable signal, and a transmitter that takes the output of the sensor and transforms it into a signal strong enough to send to the controller. These two devices are also known as the primary and secondary elements, respectively. The controllers, with a certain control algorithm or strategy, decide what to do to maintain the controlled variable at the desired value. Based on this decision, the control signal is sent to the final control actuator to manipulate the process. Depending on how the controller uses the information, the automatic process control can be further divided into three types, open-loop, closed-loop, and combined controls, as shown in Fig. 2.2.

The open-loop system does not use the measurement of the controlled variable or any disturbances, and this kind of control is only applicable in cases with very good prior knowledge of the controlled process and a lack of disturbances; both cannot be satisfied in injection molding process control. In the closed-loop control, the controller receives the signal of the measurement unit and compares it with the desired values to make the control decision; for example, the u in Fig. 2.2(b) is based on the observations of process output y. This kind of closed-loop control is also known as feedback control: the current control decisions are made based on the observations of the effects of former decisions.

It is necessary to note that in the closed-loop control scheme, there is no need for a complete and precise knowledge of the controlled process. The additional information of the disturbance loop of how d affects u can be obtained via the measurement of process output effects. Furthermore, all of the disturbances will eventually have

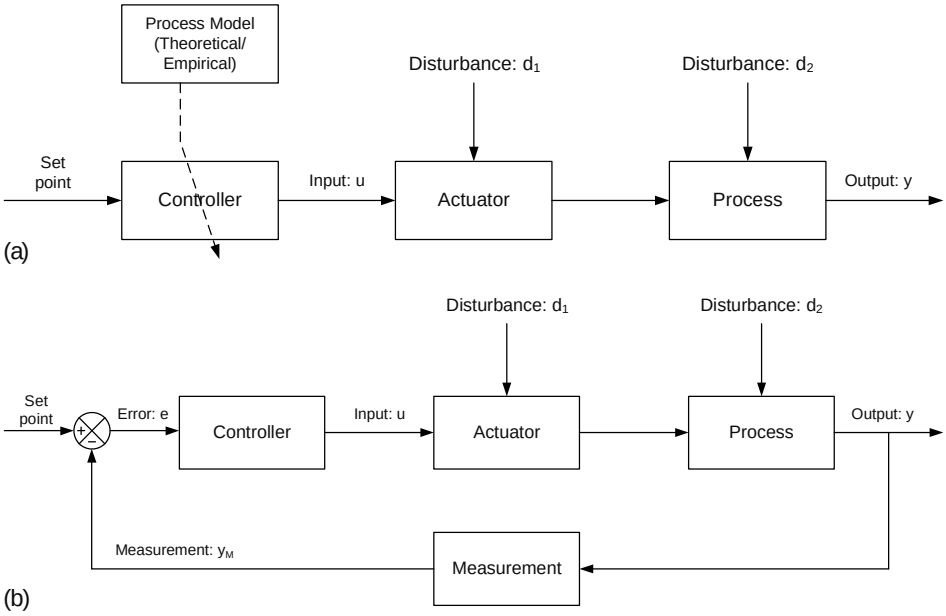

Figure 2.2 Illustration of open-loop (a) and closed-loop (b) controls

certain influences on the output, and these influences can be compensated for by the feedback mechanism. However, in the closed-loop system, the control action counteracting the disturbances always lags behind the disturbance loop, and an open-loop system with measurement of the disturbance could be helpful in this kind of situation. Later in this book the combined control structure to improve control performance is presented.

2.2 Traditional Feedback Control: PID

The proportional-integral-derivative (PID) controller was first developed for automatic ship steering, and it is the most standard feedback control algorithm that measures the controlled variable, calculates the error between the output and set point, and generates the controller output based on the proportional, the integral, and the derivative of the errors. The controller uses not only the current error, but also the past error and the current rate of change. The original form of the continuous PID control algorithm can be written as follows:

$$u(t) = K_P e(t) + t_I \int_0^t e(t)\,dt + t_D \frac{d}{dt} e(t) \tag{2.1}$$

where $e(t) = y_{set}(t) - y_m(t)$ is the control error at time t, t is the instantaneous time, τ is the variable of integration, and K_P, τ_I, and τ_D are the proportional gain, integral gain, and derivative gain, respectively. In Eq. 2.1, $K_P e(t)$, $\tau_I \int_0^t e(\tau)\,d\tau$, and $\tau_D \frac{d}{dt} e(t)$ are the proportional term, integral term, and derivative term, respectively. The early PID controller was mostly implemented using analog circuits, but with the rapid development of computer-based control technology, the digital or discretized PID control has become widely used in modern control engineering:

$$u(k) = K_P e(k) + K_I \sum_{i=1}^{k} e_i + K_D \left[e(k) - e(k-1) \right] \tag{2.2}$$

The original positon form of PID control as formulated in Eqs. 2.1 and 2.2 could cause some problems, such as integration windup and difficulty in doing the integration and derivative. A more general expression is often used in industry, known as the velocity form:

$$\begin{aligned} u(k) &= u(k) - u(k-1) \\ &= K_P \left[e(k) - e(k-1) \right] + K_I e(k) + K_D \left[e(k) - 2e(k-1) + e(k-2) \right] \end{aligned} \tag{2.3}$$

The velocity form of PID control algorithm has some advantages. In particular, it inherently removes the integration windup problem because the integration term is eliminated in Eq. 2.3.

As a simple and general-purpose control algorithm, the PID may be the most successful automatic controller in industry. For the injection molding process, PID control is commonly used in the barrel and mold temperature control. In the early practice, it was also used to control some key process variables, such as injection velocity and packing pressure. However, it also has some significant limitations. For example, it only works for linear and time-invariant processes and is not suitable for complex and nonlinear processes like injection molding, and the parameters of a PID controller are fixed, so it cannot be used as the core of an advanced control system. The PID control is suitable for continuous processes because, for a continuous process normally working around a certain operating point, the process dynamics can be linearized in a small range, and PID control can be effective under such a circumstance. As a typical batch process, the injection molding is stage-based and often operating over a wide range of conditions, and the traditional fixed parameter controller cannot ensure a satisfactory performance.

To illustrate the performance of a PID controller in an injection molding process application, a PID injection velocity control is presented below. The control experiments were conducted on an industrial injection molding machine with clamping tonnage of 88 tons, and the hydraulic power system was controlled by a fast-response

servo-valve. For easy description and without loss of generality, a PI controller was designed and tuned for the injection velocity using the Ziegler-Nichols (Z-N) method [17] near the working point of 50 mm/s. The first step of the Z-N method is to experimentally determine the ultimate gain K_{cu}. The period of the resulting sustained oscillation is referred to as the ultimate period P_u. Then K_P and K_I can be calculated using the relations $K_P = 0.45K_{cu}$ and $K_I = K_P \frac{\Delta t}{P_u/1.2}$, where Δt is the sampling period. For this specific problem, the parameters were determined to be $\Delta t = 5$ ms, $K_P = 50$, and $K_I = 4.167$.

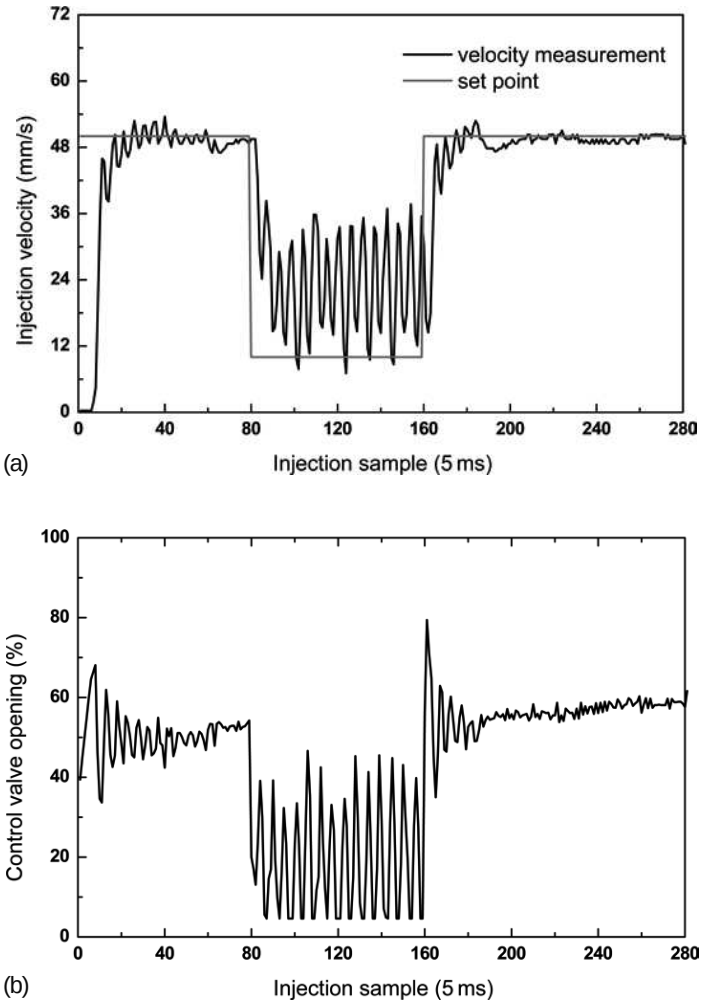

Figure 2.3 PI controller response to a two-step change set-point profile: (a) the injection velocity response, and (b) the servo-valve opening

Figure 2.3 shows the PI controller response to a step-change velocity profile. The controller works well near the tuning point. However, when the set point changes from 50 to 10 mm/s, the PI control response oscillated strongly, due to the process nonlinearity. The oscillation of the response may be reduced by slowing down the response time of the PI controller, but a slow response is obviously undesirable in injection velocity control. The poor performance of the PI controller not only confirmed the nonlinear characteristic of the injection velocity, but also proved the necessity of using advanced control algorithms.

2.3 Adaptive Control

It was shown in the previous section that the traditional fixed-parameter control algorithm such as PID control cannot work well for the injection molding process, due to the batch operation nature and the nonlinear and time-varying characteristics of the process. Adaptive control is a good alternative for this kind of control. Its parameters are adapted in a certain way to conform to the nonlinear or time-varying process dynamics and provide a good control performance. There are many different types of adaptive control schemes, such as gain scheduling, model reference adaptive control, dual adaptive control, and self-tuning regulators (STR). The STR, as an important scheme of adaptive control, is used for illustration in this book to control some key process variables in injection molding. The basic principle of STR is briefly described in the following sections, and detailed discussions can be found in references [18–20].

A self-tuning system is graphically shown in Fig. 2.4 [14]. The system is composed of two loops: an ordinary feedback-control loop, as shown with the dashed line, and a controller parameter-adjusting loop, as shown with the dotted-line block. The latter, consisting of a parametric model estimator and a controller design calculator, gives an online adjustment of the parameters of the feedback controller. The process model parameters and controller design are updated during each sampling period, with a specified model structure.

There are several methods for process model parameter estimation, for example, least mean squares (LMS), projection algorithm (PA), and stochastic approximation (SA). In this book, a recursive least-squares (RLS) estimator is used because of its good sensitivity and superior convergence property [19]. A simple but effective pole-placement control design is adopted for the controller design first to demonstrate the working procedure of the STR.

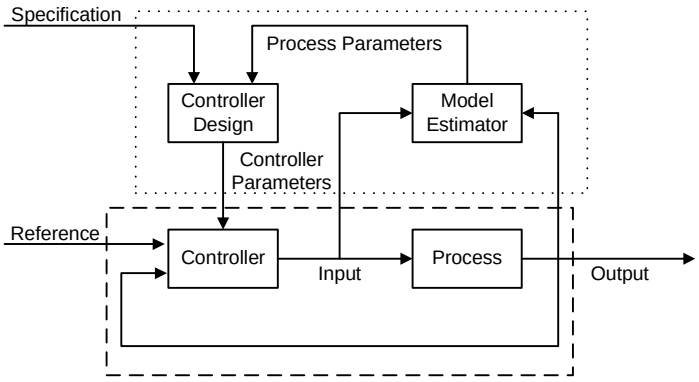

Figure 2.4 Block diagram of an adaptive self-tuning regulator

2.3.1 Model Estimation

Assuming that the process dynamics may be modeled by a discrete time autoregressive with external input (ARX) model, we have

$$A(z)\,y(t) = B(z)\,u(t - n_d) + e(t) \tag{2.4}$$

where

$$A(z) = 1 + a_1 z^{-1} + \cdots + a_{n_a} z^{-n_a}$$

$$B(z) = \left(b_0 + b_1 z^{-1} + \cdots + b_{n_b-1} z^{-n_b+1}\right) \times z^{-n_d}$$

u: inputs to the process

y: corresponding observed process outputs

z: z-transform (time-shift) operator

n_a, n_b, and n_d: the orders of A, B polynomials and process delay, respectively.

Introduce the process model parameter vector:

$$\theta^T = \begin{bmatrix} a_1 & \cdots & a_{n_a} & b_0 & \cdots & b_{n_b-1} \end{bmatrix} \tag{2.5}$$

the regression vector:

$$\varphi^T(t) = \begin{bmatrix} y(t-1) & \cdots & y(t-n) & u(t-n_d) & \cdots & u(t-n_d-m+1) \end{bmatrix} \tag{2.6}$$

and the loss function:

$$V(\theta, t) = \frac{1}{2} \sum_{i=1}^{t} \left[y(i) - \varphi^T(i)\theta \right]^2 \tag{2.7}$$

The model parameter q, which minimizes $V(\mathrm{q},t)$, the differences between the output observation, $y(i)$, and its prediction, $\mathrm{j}^{\mathrm{T}}(i)\mathrm{q}$, in the least-squares sense, is given recursively by

$$\mathrm{q}(t) = \mathrm{q}(t-1) + K(t)\left[y(t) - \mathrm{j}(t)^{\mathrm{T}}\mathrm{q}(t-1)\right] \tag{2.8}$$

$$K(t) = P(t-1)\mathrm{j}(t)\left[I + \mathrm{j}(t)^{\mathrm{T}}P(t-1)\mathrm{j}(t)\right]^{-1} \tag{2.9}$$

$$P(t) = \left[I - K(t)\mathrm{j}^{\mathrm{T}}(t)\right]\frac{P(t-1)}{\lambda} \tag{2.10}$$

The variable λ in Eq. 2.9 is a forgetting factor that dictates how fast the model is updated. The value of λ is $0 < \lambda \leq 1$. The smaller λ is, the faster the estimator can track the model changing, and a small λ will also make the estimation more sensitive to measurement noises. In this project, λ is set to be 0.98 for injection velocity control and 0.99 for packing pressure control because the selections produce good estimates. As a rule of thumb, the estimate is based on the last N-step results, and N can be calculated as below [19]:

$$N = \frac{2}{1-\lambda} \tag{2.11}$$

2.3.2 Pole-Placement Controller Design

A pole-placement design is adopted here as the feedback controller in STR for demonstration purposes, as shown in Fig. 2.5 [14]. The design is to find a controller that gives a closed-loop system response specified by the desired closed-loop pole locations. With the process described by Eq. 2.4, the controller has one output, u, and two inputs, the command signal, u_C, and the measured process output, y. A linear controller relating the output to its inputs may be expressed by

$$R(z)u(k) = T(z)u_{\mathrm{C}}(k) - S(z)y(k) \tag{2.12}$$

where $R(z)$, $S(z)$, and $T(z)$ are controller parameter polynomials.

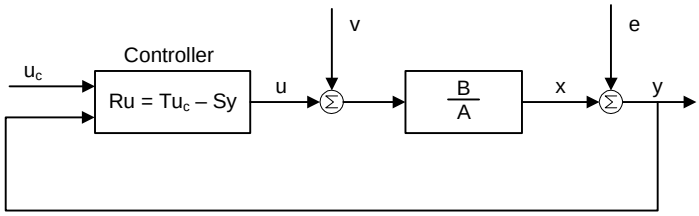

Figure 2.5 Structure of pole-placement feedback controller

The goal of the controller design is to find these three polynomials for the given closed-loop poles. It can be derived that the closed-loop characteristic polynomial is [19]

$$AR + BS = A_C. \tag{2.13}$$

The desired closed-loop characteristic polynomial A_C may be specified as

$$A_C = A_1 z^d = \left(z^2 + p_1 z + p_2\right) z^d \tag{2.14}$$

where $p_1 = -2e^{-\zeta\omega_0 h}\cos\left(\omega_0 h\sqrt{1-\zeta^2}\right)$, and $p_2 = e^{-2\zeta\omega_0 h}$.

The parameter A_1 is equivalent to the second-order system characteristic equation in the s domain:

$$A_1(s) = s^2 + 2\zeta\omega_0 s + \omega_0^2 \tag{2.15}$$

Here z^d is a deadbeat observer with all eigenvalues equal to zero. The damping factor, ζ, and natural frequency, ω_0, decide the closed-loop response of the system. A value of ζ larger than 1.0 yields a sluggish response, while $0 < \zeta < 1.0$ makes the responses exhibit oscillation and overshoot. The fastest response without overshoot is obtained for the critical damping factor, $\zeta = 1.0$. In terms of pole locations, a bigger ω_0 makes the desired poles closer to the origin in the z plane, which leads the response of the system to be faster but more sensitive to the model error. A smaller ω_0, however, yields a slower response that is less sensitive to the model error and signal noise. Once the desired closed-loop characteristic equation is determined, the controller polynomials $R(z)$, $S(z)$, and $T(z)$ can be obtained by solving Eq. 2.9, which is known as the polynomial Diophantine equation.

2.3.3 Solving the Diophantine Equation

The most important part of the controller design is to solve the Diophantine Eq 2.12. There are several alternative methods to solve this equation [19]. The first is to equate the coefficients of the right and left sides of the equation. This method works only for low-order systems, and when the model order is changed, the equation has to be rearranged and solved again. The second alternative, called the Sylvester matrix method, is to use a matrix calculation, which is computationally inefficient and not suitable for online implementation. The Euclidean method, a polynomial method based on the extended Euclidean algorithm, is a good selection in this application as it is the most efficient method to solve the equation. This method works for a wide range of model orders since it exploits the structure of the problem [19]. The Diophantine equation is solved using the extended Euclidean algorithm through

two steps [23]. The first step is to find four polynomials X, Y, U, and V that satisfy the following equations:

$$AX + BY = G \tag{2.16}$$

$$AU + BV = 0 \tag{2.17}$$

where G is the greatest common divisor of A and B. If A and B were relatively prime, G equals one. The generalized Euclid's algorithm is used to find polynomials X, Y, U, and V, as detailed in reference [21]. With the solutions of Eq. 2.20 and 2.21, the method proceeds to step two to find the minimum-degree solution for Eq. 2.12. The particular solution is given by Eqs. 2.17 and 2.18:

$$R_0 = XA_C \operatorname{div} G \tag{2.18}$$

$$S_0 = YA_C \operatorname{div} G \tag{2.19}$$

where div is the polynomial division operator.

The general solution is given by Eqs. 2.19 and 2.20:

$$R = R_0 + QU \tag{2.20}$$

$$S = S_0 + QV \tag{2.21}$$

where Q is an arbitrary polynomial. It has been proved that there exists a unique solution to Eq. 2.9 such that $\deg X < \deg B$ or $\deg Y < \deg A$. This solution with $\deg Y < \deg A$ is obtained from Eq. 2.20 by choosing $Q = - S_0 divV$. Thus, the solutions to the Diophantine Eq. 2.12 are given by Eqs. 2.22–2.24:

$$Q = - S_0 \operatorname{div} V \tag{2.22}$$

$$R = R_0 + QU \tag{2.23}$$

$$S = S_0 \bmod V \tag{2.24}$$

The above procedure is theoretically suitable for all model orders. In industrial implementation, however, the model order is always limited to reduce computation resource requirements, and the process dynamics of injection molding are unlikely to be too high.

2.3.4 Injection Velocity Adaptive Control Result

The adaptive controller described previously is applied to the injection velocity control again on the 88-ton molding machine with a hydraulic power system

4 Two-Dimensional Control Algorithms

4.1 Two-Dimensional Control Background

Injection molding is a typical batch process; it has its own characteristics in comparison to a continuous process. The obvious differences between a continuous process and a batch process like injection molding are (1) a batch process has a finite duration, (2) a batch process repeats itself until the specified amount of product has been made, and (3) a batch process is processed by an ordered set of activities. These characteristics make the control schemes proposed for a continuous process ill-suited for injection molding. Modifications of the original control algorithms have to be made to cope with these features. To summarize the difference between injection molding and a traditional continuous process, the distinctive nature of an injection molding process has three aspects:

1. Repetitive nature: the injection molding process repeats itself batch to batch to produce the same products;
2. Two-dimensional (2D) dynamic nature: there are within-batch and batch-to-batch dynamics in injection molding simultaneously; and
3. Multiphase nature: an injection molding process consists of more than one phase.

In this chapter, these aspects are studied extensively to introduce a new group of control algorithms developed specifically for batch processes like injection molding, that is, the 2D control algorithms.

Consider a digital injection molding controller in which for each cycle the control samples form a time series: if these different cycles' (with cycle duration T and cycle number K) information is aligned as shown in Fig. 4.1, a two-dimensional system can be formed. Although cycle-wise dynamics do exist, for example, during the warm-up stages when the machine is slowly reaching steady operation, these different cycles' time series still lack a cycle direction causality due to the independent operation of each cycle.

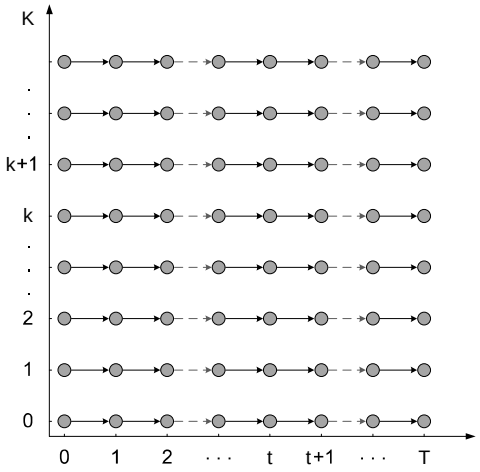

Figure 4.1 Alignment of different cycles' information

Consider an injection molding process controlled by ILC. Any cycle's system output depends not only on the control action and output of the current cycle, but also the information from the last cycle. The cycle-wise dynamic is naturally introduced, and the 2D system becomes a causal system, as schematically shown in Fig. 4.2.

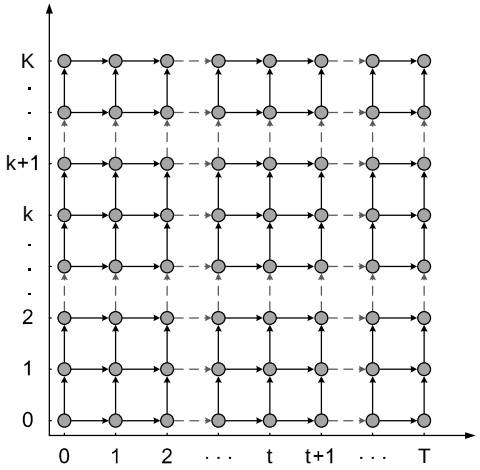

Figure 4.2 Schematic of 2D causal system

The traditional 2D system and its analysis are designed for time-independent series, such as image processing. For this kind of 2D system, at current point $P(k,t)$, all of the information about points $P(i,j), i < k \text{ È } j < t$, is known. The 2D system constructed with the ILC-controlled injection molding process is time dependent: only information about previous cycles or the current cycle's previous sampling periods

is known. Hence, the causality, controller design, and analysis of this time-dependent 2D system are totally different from those of the traditional one. To clearly show this difference, Fig. 4.3 plots the information-acquisition sequence of these systems, where Fig. 4.3(a) is the traditional 2D system and Fig. 4.3(b) is the 2D system of an injection molding process.

In this chapter, a 2D generalized, predictive, iterative learning control (2D-GPILC) based on a 1D model is introduced first, followed by a 2D dynamic matrix control (2D-DMC) also based on a 1D model. The system identification of the 2D model is applied, followed by a 2D generalized predictive control (2D-GPC) based on the 2D model.

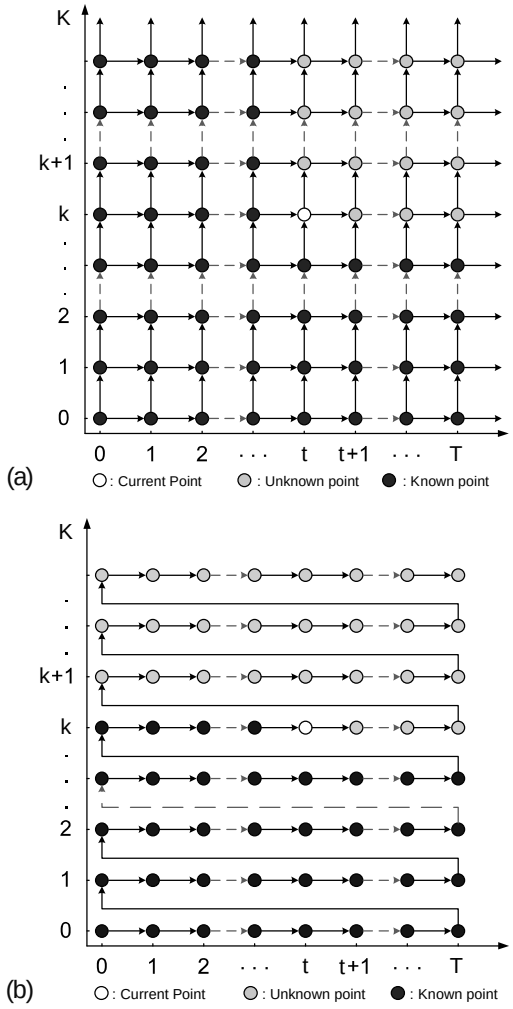

Figure 4.3 Comparison of information-acquisition sequence:
(a) traditional 2D system, and
(b) 2D system of an injection molding process controlled by ILC

4.2 Two-Dimensional Generalized Predictive Iterative Learning Control

4.2.1 2D-GPILC Control Algorithm

The control algorithm is introduced briefly here. The detailed derivation and analysis were given in references [64–67].

A SISO batch process can be described by the following controlled autoregressive integrated moving-average (CARIMA) model:

$$S_P: \quad A\left(q_t^{-1}\right) y_k(t) = B\left(q_t^{-1}\right) D_t\left(u_k(t)\right) + w_k(t) \tag{4.1}$$

$t = 0, 1, \square, T; \quad k = 1, 2, \square$

where t and k represent the discrete-time and cycle/batch index, respectively; T is the time duration of each cycle; $u_k(t)$, $y_k(t)$, and $w_k(t)$ are the input, output, and disturbance of the process at time t in the kth cycle, respectively; q_t^{-1} indicates the time-wise unit backward-shift operator; $A\left(q_t^{-1}\right)$ and $B\left(q_t^{-1}\right)$ are both operator polynomials:

$$A\left(q_t^{-1}\right) = 1 + a_1 q_t^{-1} + a_2 q_t^{-2} + \square + a_n q_t^{-n} \tag{4.2}$$

$$B\left(q_t^{-1}\right) = b_1 q_t^{-1} + b_2 q_t^{-2} + \square + b_m q_t^{-m} \tag{4.3}$$

and D_t represents the time-wise backward difference operator, or $D_t\left(f(t,k)\right) = f(t,k) - f(t-1,k)$.

For the above repetitive process, introduce an ILC law with the form

$$S_{ILC}: \quad u_k(t) = u_{k-1}(t) + u_k(t-1) - u_{k-1}(t-1) + r_k(t) \tag{4.4}$$

$u_0(t) = 0, \quad t = -1, 0, 1, \square, T$

where $r_k(t)$ is referred to as the updating law, to be determined online based on the MPC philosophy, and $u_0(t)$ is the initial profile of iteration. Letting q_k^{-1} represent the cycle-wise unit backward-shift operator, $u_k(t)$ and $r_k(t)$ have the following relationship:

$$u_k(t) = \frac{1}{\left(1 - q_k^{-1}\right)} \times \frac{1}{\left(1 - q_t^{-1}\right)} \times r_k(t) \tag{4.5}$$

which is a 2D system with 2D integral transformation.

Control law Eq. 4.4 can be expressed equivalently as

$$\mathrm{D}_t\left(u_k(t)\right) = \mathrm{D}_t\left(u_{k-1}(t)\right) + r_k(t) \tag{4.6}$$

or

$$\mathrm{D}_k\left(u_k(t)\right) = \mathrm{D}_k\left(u_k(t-1)\right) + r_k(t) \tag{4.7}$$

where D_k represents the cycle-wise backward difference operator, or $\mathrm{D}_k\left(f(t,k)\right) = f(t,k) - f(t,k-1)$.

Substituting Eq. 4.7 into Eq. 4.1 leads to the following 2D system:

$$\mathrm{S}_{2D\text{-}P}: \quad A\left(q_t^{-1}\right)y_k(t) = A\left(q_t^{-1}\right)y_{k-1}(t) + B\left(q_t^{-1}\right)r_k(t) + \mathrm{D}_k\left(w_k(t)\right) \tag{4.8}$$

Here $r_k(t)$, $y_k(t)$, and $\mathrm{D}_k\left(w_k(t)\right)$ represent the input, output, and disturbance of the system, respectively.

To ensure smooth operations along both the time and cycle directions, the following quadratic cost function is defined as the general control performance index over one cycle for the 2D system of Eq. 4.8:

$$\begin{aligned} J(t,k,n_1,n_2) &= \begin{bmatrix} \hat{\mathrm{e}}_{k|k}\left(\left|{t+1 \atop t+n_1}\right|t\right) \\ \mathrm{D}_t\left(\mathrm{u}_k\left(\left|{t \atop t+n_2-1}\right.\right)\right) \\ \mathrm{D}_k\left(\mathrm{u}_k\left(\left|{t \atop t+n_2-1}\right.\right)\right) \end{bmatrix}^T \begin{bmatrix} Q & 0 & 0 \\ 0 & S & 0 \\ 0 & 0 & T \end{bmatrix} \begin{bmatrix} \hat{\mathrm{e}}_{k|k}\left(\left|{t+1 \atop t+n_1}\right|t\right) \\ \mathrm{D}_t\left(\mathrm{u}_k\left(\left|{t \atop t+n_2-1}\right.\right)\right) \\ \mathrm{D}_k\left(\mathrm{u}_k\left(\left|{t \atop t+n_2-1}\right.\right)\right) \end{bmatrix} + \mathrm{r}_k^T\left(\left|{t \atop t+n_2-1}\right.\right)\mathrm{R}r_k\left(\left|{t \atop t+n_2-1}\right.\right) \\ &= \hat{\mathrm{X}}_{k|k}^T(t)\bar{\mathrm{Q}}\hat{\mathrm{X}}_{k|k}(t) + \mathrm{r}_{k+l}^T\left(\left|{t \atop t+n_2-1}\right.\right)\mathrm{R}r_{k+l}\left(\left|{t \atop t+n_2-1}\right.\right) \end{aligned} \tag{4.9}$$

where the notations are

$$\begin{aligned} \mathrm{f}_k\left(\left|{t_1 \atop t_2}\right.\right) &= \left(f_k(t_1) \quad f_k(t_1+1) \quad \square \quad f_k(t_2)\right) \\ f &\in \{y,u,r,e\} \end{aligned} \tag{4.10}$$

and $\hat{\mathrm{f}}_{k|k}\left(\left|{t+1 \atop t+n_1}\right|t\right)$ indicates the prediction vector of the kth cycle over the time-wise prediction horizon based on the history information before time t of the kth cycle, and

$$\hat{\mathrm{e}}_{k|k}\left(\left|{t+1 \atop t+n_1}\right|t\right) = \mathrm{y}_r\left(\left|{t+1 \atop t+n_1}\right.\right) - \hat{\mathrm{y}}_{k|k}\left(\left|{t+1 \atop t+n_1}\right|t\right) \tag{4.11}$$

$$\hat{\mathrm{X}}_{k|k}(t) = \begin{bmatrix} \hat{\mathrm{e}}_{k|k}\left(\left|{t+1 \atop t+n_1}\right|t\right) \\ \mathrm{D}_t\left(\mathrm{u}_k\left(\left|{t \atop t+n_2-1}\right.\right)\right) \\ \mathrm{D}_k\left(\mathrm{u}_k\left(\left|{t \atop t+n_2-1}\right.\right)\right) \end{bmatrix} \tag{4.12}$$

5.3 Multivariate Statistical Analysis Methods for SPM

Principal component analysis (PCA) and partial least squares (PLS) are the most popular multivariate statistical approaches for process monitoring, which will be extensively studied and applied to an injection molding process.

5.3.1 Principal Component Analysis and Partial Least Squares

Given a two-dimensional data matrix $X \in R^{n \times m}$, where n is the number of samples and m is the number of process variables, PCA decomposes X as

$$X = TP^T = \sum_{j=1}^{m} t_j p_j^T = t_1 p_1^T + t_2 p_2^T + \cdots + t_m p_m^T \tag{5.1}$$

where $t_j \in R^{n \times 1}$ is the latent variable vector, also called the principal component (PC) vector or score vector; $p_j \in R^{m \times 1}$ is the loading vector that can project the original process data into the score space and contain the variables' correlation information; T and P are the score matrix and loading matrix, respectively. In the decomposition, scores are made to be orthogonal to each other, which means $t_i^T t_j = 0$ when $i \neq j$. At the same time, the loading vectors are orthogonal to each other, satisfying $p_i^T p_j = 0$ when $i \neq j$ and $p_i^T p_j = 1$ when $i = j$. Therefore,

$$t_j = Xp_j \tag{5.2}$$

$$T = XP \tag{5.3}$$

Algebraically, $\|t_j\|$ is equal to the jth largest eigenvalue λ_j of the covariance matrix $S = \frac{1}{n-1} X^T X$, and p_j is the corresponding eigenvector. Note that $\|t_1\| > \|t_2\| > \cdots > \|t_m\|$ because of $\lambda_1 > \lambda_2 > \cdots > \lambda_m$. It is easy to understand that the first several PCs contain the most variance information of X, and the last few PCs may contain only noises. This indicates that the majority of variance information can be extracted by retaining the first few orthogonal PCs, and the dimension of original variables can be largely reduced accordingly.

By retaining only the first A PCs, X can be approximated by

$$\hat{X} = \overline{T}\overline{P}^T = \sum_{j=1}^{A} t_j p_j^T \tag{5.4}$$

$$\overline{T} = X\overline{P} \tag{5.5}$$

$$\hat{X} = X\overline{P}\overline{P}^T \tag{5.6}$$

where $\bar{T} \in R^{n \times A}$ is the score matrix in PC subspace, $\bar{P} \in R^{m \times A}$ is the corresponding loading matrix, and $\hat{X} \in R^{n \times m}$ is the reconstruction of the original data matrix X. Equation 5.4 interprets the PCA model from the aspect of matrix decomposition, while Eq. 5.6 is from the aspect of data reconstruction. The loading matrix $\bar{P}$ is the core, which can be considered as the parameter matrix of a PCA model.

Then,

$$X = \hat{X} + E \tag{5.7}$$

$$E = \tilde{T}\tilde{P}^T = \sum_{j=A+1}^{m} t_j p_j^T \tag{5.8}$$

where $E \in R^{n \times m}$ is the residual matrix, $\tilde{T} \in R^{n \times (m-A)}$ is the score matrix in residual subspace, and $\tilde{P} \in R^{m \times (m-A)}$ is the corresponding loading matrix. From Eqs. 5.4, 5.5, and 5.8, it is clear that the PC subspace spanned by $\bar{P}$ and the residual subspace spanned by $\tilde{P}$ are orthogonal to each other.

Several algorithms, such as singular value decomposition (SVD) and nonlinear iterative partial least squares (NIPALS), can be used to calculate the loading matrix [91]. There are many methods to determine the number of retained PCs, among which the cross-validation method [92] is the most popular.

Different from PCA, PLS works on two data matrices, the process data matrix $X \in R^{n \times m_x}$ and the product quality data matrix $Y \in R^{n \times m_y}$, where n is the number of samples, m_x is the number of process variables, and m_y is the number of quality variables. PLS extracts the variation of X and gives as much prediction to Y as possible at the same time.

The PLS model can be formulated as

$$X = \bar{T}\bar{P}^T + E = \sum_{j=1}^{A} t_j p_j^T + E \tag{5.9}$$

$$Y = \bar{U}\bar{Q}^T + F = \sum_{j=1}^{A} u_j q_j^T + F \tag{5.10}$$

$$u_j = b_j t_j + e \tag{5.11}$$

Equations 5.9 and 5.10 describe the inner projection structures in X and Y, where $\bar{T}$ and $\bar{P}$ are the score matrix and loading matrix decomposed from X; $\bar{U}$ and $\bar{Q}$ are the score matrix and loading matrix decomposed from Y; and E and F are the residual matrices. Equation 5.11 describes the outer projection structure between X and Y, where $b_j = t_j^T u_j / (t_j^T t_j)$ is the regression coefficient between the latent variable t_j derived from X and the latent variable u_j derived from Y.

The PLS model given by Eqs. 5.9–5.11 can be written in a compact way as

$$Y = XQ^T + F^* \tag{5.12}$$

where Q is the regression parameter matrix and F^* is the residual matrix.

The detailed properties, the calculation methods of PLS (e.g., NIPALS), and the methods to choose the number of latent variables (cross-validation, jackknife, and so on) can be found in the references [93–95]. In addition, Dayal and MacGregor [96] have developed another algorithm called Kernel PLS to compute the model parameters.

5.3.2 PCA/PLS-Based Statistical Process Monitoring

After PCA modeling, the loading matrix $\overline{P}$ is obtained based on the historical process data. When online process monitoring is conducted, a new sample $x_{new} = [x_{new,1}, \ldots, x_{new,m}]$ is measured and projected onto the score subspace using the loading matrix $\overline{P}$ to get its score vector t_{new} and the residual vector e_{new} by

$$t_{new} = x_{new}\overline{P} \tag{5.13}$$

$$\hat{x}_{new} = t_{new}\overline{P}^T = x_{new}\overline{P}\overline{P}^T \tag{5.14}$$

$$e_{new} = x_{new} - \hat{x}_{new} = x_{new}(I - \overline{P}\overline{P}^T) \tag{5.15}$$

where I is an $m \times m$ identity matrix.

For process monitoring, two multivariate statistics are calculated: the Hotelling T^2 statistic and the SPE (or Q) statistic.

The Hotelling T^2 summarizes the systematic variation information extracted in score variables:

$$T^2 = t_{new}S^{-1}t_{new}^T = \sum_{i=1}^{A} \frac{t_{new,i}^2}{\lambda_i} \tag{5.16}$$

where $t_{new} = [t_{new,1}, \ldots, t_{new,A}]$, $t_{new,a}$ $(a = 1,\ldots,A)$ is the ath score variable, $S = diag(\lambda_1, \ldots, \lambda_A)$, λ_a is the ath largest eigenvalue of the covariance matrix $S = \frac{1}{n-1}X^TX$, and X is the normal history data used in PCA modeling.

Taking the assumption that scores of normal process data obey a multivariate normal distribution, the control limits of T^2 can be calculated using F distribution:

$$T^2 \sim \frac{A(n-1)}{n-A}F_{A,n-A,\alpha} \tag{5.17}$$

where a is the significance level, and $F_{A,n-A,\alpha}$ is the critical value of F distribution with significance level of a and degrees of freedom of A and $n-A$, whose value can be found in a statistical table.

The SPE statistic measures the projection of a sample vector onto the residual subspace:

$$\mathrm{SPE} = \mathbf{e}_{new}\mathbf{e}_{new}^{T} = \sum_{j=1}^{m} (x_{new,j} - \hat{x}_{new,j})^2 \tag{5.18}$$

where $x_{new,j}$ is the jth element in $\mathbf{x}_{new}$, and $\hat{x}_{new,j}$ is the corresponding reconstructed value using the developed PCA model. The SPE statistic measures variability that breaks the normal process correlation, which often indicates an abnormal situation.

With the assumption that residuals are normally distributed when the process is fault free, the control limits of SPE can be derived as [91]:

$$\mathrm{SPE}_{\alpha} = \theta_1 \left[\frac{C_{\alpha}\sqrt{2\theta_2 h_0^2}}{\theta_1} + 1 + \frac{\theta_2 h_0 (h_0 - 1)}{\theta_1^2} \right]^{\frac{1}{h_0}} \tag{5.19}$$

where $\theta_i = \sum_{j=A+1}^{m} \lambda_j^i \quad (i = 1,2,3)$, $h_0 = 1 - \frac{2\theta_1\theta_3}{3\theta_2^2}$, and C_{α} is the critical value of the normal distribution under the significance level of a .

The Hotelling T^2 statistic measures the distance between $\mathbf{t}_{new}$ and the origin of historical normal scores in $\overline{T}$. If a sample exceeds the T^2 limit, it does not break the correlation structure but shifts away from the normal PC subspace. In process monitoring, the values of these two statistics are plotted in the Hotelling T^2 and SPE control charts and compared with the corresponding control limits. If any of them exceeds the control limits, the process is considered to be out of control, and a fault is alarmed.

Like PCA, the above T^2 and SPE statistics are also commonly applied for PLS-based process monitoring. But, more often, PLS is used for online quality prediction, which will be introduced in Chapter 7.

After a fault has been detected by the T^2 or SPE control chart, it is desirable to find out the cause of the fault. A contribution plot [97] is widely applied, which shows the contribution of each process variable to the detected fault using a bar chart. Variables with large contributions need to be investigated since they are affected by the fault significantly. It should be noted that a contribution plot cannot diagnose the fault cause definitely. Process knowledge is always required to assist the contribution plot for fault identification.

The contribution of $x_{new,j}$ to the ath PC score $t_{new,a}$ can be derived as [98],

$$C_{t_{new,a},x_{new,j}} = x_{new,j} p_{a,j} \Big/ t_{new,a} \tag{5.20}$$

where $p_{a,j}$ is the ath element in the jth column in the loading matrix P. The confidence limits for contribution plots can be used to identify a set of faulty variables [99].

For SPE, the contribution of $x_{new,j}$ is given as

$$C_{SPE,x_{new,j}} = sign(x_{new,j} - \hat{x}_{new,j}) \times \frac{(x_{new,j} - \hat{x}_{new,j})^2}{\text{SPE}} \tag{5.21}$$

5.3.3 Multiway PCA/PLS

PCA and PLS can only deal with two-dimensional data matrices. For an injection molding process that is a typical batch process, process data are usually represented by a three-dimensional data matrix $\underline{X} \in R^{I \times K \times J}$, where I is the number of normal batches, J is the number of process variables, and K is the number of total sampling intervals in a batch. To apply multivariate statistical methods like PCA or PLS for process monitoring, a three-dimensional data matrix should be transformed into a two-dimensional data matrix. Figure 5.5 shows two meaningful ways of matrix unfolding.

Figure 5.5(a) is called batch-wise unfolding. It keeps the dimension of batches, and merges variable and time dimensions, where $\bar{X}^k$ is the time-slice data matrix of the kth sampling time. Each row of the unfolded two-dimensional matrix $X \in R^{I \times JK}$ contains all data within a batch. Different from batch-wise unfolding, variable-wise unfolding, as shown in Fig. 5.5(b), keeps the dimension of variables and merges the other two dimensions, generating a two-dimensional matrix $X \in R^{KI \times J}$. Each sampling point of each batch is considered as an object. Batch-wise unfolding is more popular than variable-wise unfolding because batch variation is the main concern in batch process monitoring.

Based on batch-wise unfolding, multiway PCA (MPCA) and multiway PLS (MPLS) were proposed [100, 102]. The basic idea of MPCA is to perform PCA on the unfolded data matrix $X \in R^{I \times JK}$, supposing X has been normalized, while MPLS does a regression between the unfolded process data matrix $X \in R^{I \times J_x K}$ and quality data matrix $Y \in R^{I \times J_y}$, where J_x is the number of process variables and J_y is the number of quality variables.

MPCA is essentially similar to PCA. The only difference lies in that MPCA is able to deal with three-dimensional process data, which makes PCA available for batch process monitoring. The main principle of MPCA can be expressed as the following three steps, which are shown in Fig. 5.6:

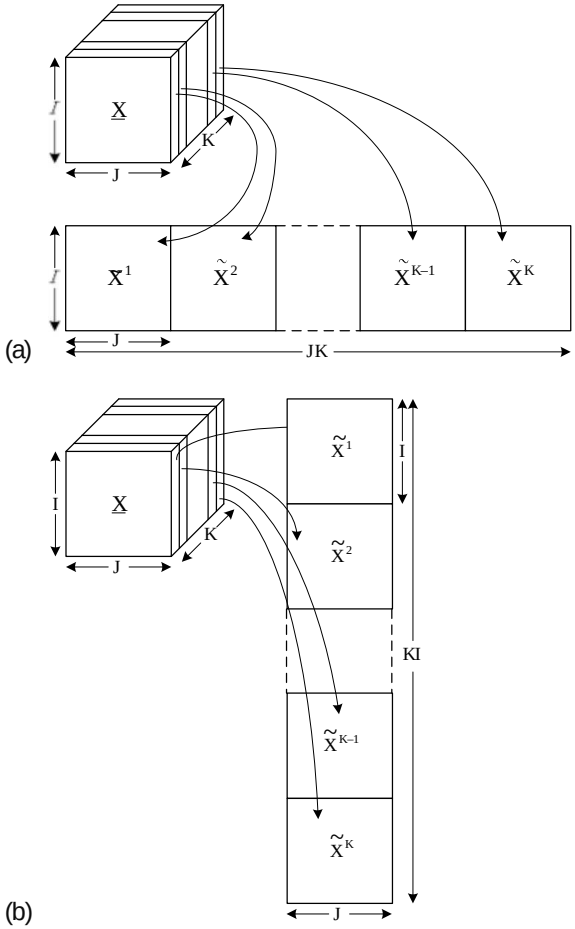

Figure 5.5 Three-dimensional matrix unfolding:
(a) batch-wise unfolding; (b) variable-wise unfolding

1. Unfold the three-dimensional data matrix $\underline{X}$ into two-dimensional data matrix X using batch-wise unfolding.
2. Calculate the score vector $t_a \in R^{I}$ and corresponding loading vector $p_a \in R^{JK}$ ($a = 1,..., A$; A is the number of retained PC).
3. Transfer the one-dimensional loading vector p_a into a two-dimensional matrix, $P_a \in R^{J \times K}$.

The MPCA model is defined as

$$\underline{X} = T_A \otimes P_A + \underline{E} = \sum_{a=1}^{A} t_a \otimes P_a + \underline{E} = \sum_{a=1}^{A} t_a p_a^T + E \qquad (5.22)$$

$$\hat{\underline{X}} = T_A \otimes P_A$$

9.3 Neural Network Model of Average-flow-length

It is not economical and practical for all molds to be fitted with a capacitive transducer as developed in the preceding section. This section discusses the model of the average-flow-length for injection velocity profiling. A first-principles model for the filling process has been under active research over the last three decades. It has been developed from simple one-dimensional models [181–197] to two-dimensional models [198–206] and more complete and complicated three-dimensional models [207–210]. The complexity of those first-principles models precludes any analytical solution. Numerical methods such as finite element methods have to be employed in solving those models. In those models, mold geometry has to be carefully modeled. This type of first-principles model is ill-suited for online prediction of melt flow behaviors. In modeling nonlinear and complex processes, a neural network approach has become a popular tool due to its fast computing and learning-by-example features. To profile the injection velocity, a soft-sensor scheme using neural networks was introduced to correlate the melt-front area (or average-flow-length) during filling with other measurable variables. With the developed capacitive transducer, a soft sensor based on the actual experimental data can be built.

9.3.1 Inputs and Output of the Neural Network Model

Different from the model based on simulation data as described in Eq. 9.5, nozzle pressure was used to replace the gate pressure as one input for the relatively convenient instrumentation at the nozzle. Furthermore, nozzle temperature, which was ignored in the simulation, was also included as an additional input; it could affect the melt viscosity and consequently the melt flow. Those changes resulted in the following relation for modeling the average-flow-length:

$$afl_n = f\left(afl_{n-1}, NP_n, \mathrm{D}NP_n, SD_n, \mathrm{D}SD_n, IV, \mathrm{D}IV, NT_n, \mathrm{D}NT_n\right) \tag{9.6}$$

where subscript n represents the nth sampling instant during filling, and symbol D stands for the variable increment during the past sampling instant. In the equation, NP_n is the nozzle pressure representing the driving force for mold filling; $\mathrm{D}NP_n$ represents the additional force required for the average-flow-length increment; SD_n is the screw displacement, which represents the total amount of melt in the mold; $\mathrm{D}SD_n$ denotes the amount of melt entering the mold during the last time increment; IV_n is the screw injection velocity, which will directly affect the melt-front velocity and consequently influence the average-flow-length; $\mathrm{D}IV_n$ is the injection velocity

increment; and NT_n and DNT_n are nozzle temperature and its increment, respectively. Injection time is not a separate input as it can be inferred from the screw displacement and injection velocity. Treated as a black box input-output relation, this model was represented by a neural network.

The neural network model has nine inputs, as listed in the right side of Eq. 9.6, and its output is the average-flow-length of the current sample. To derive a more general model and also to facilitate the neural network learning, all of the input and output variables have been normalized to be in a range of 0–1. The neural network output becomes the normalized average-flow-length, which represents the relative flow length in percentage. Similar is the screw displacement input, SD_n. Divided by the injection stroke required for the mold filling, it is also normalized in a percentage form. The other inputs, such as the nozzle pressure, nozzle temperature, and injection velocity, are divided by their maximums provided by the injection molding machine during filling. After normalization, all of the input and output variables become dimensionless.

9.3.2 Architecture of the Neural Network Model

Architecturally, neural networks can be categorized into two types [211]: feed-forward and recurrent.

The feed-forward neural network, shown in Fig. 9.7(a), is widely used, typically consisting of three types of layers: input layer, hidden layer, and output layer. The neurons in each layer of a feed-forward network have only the output signals of the preceding layer as their inputs. The nodes in the input layer supply the input signals to the nodes in the second layer (the hidden layer). The outputs of the second layer act as inputs to the third layer (the output layer).

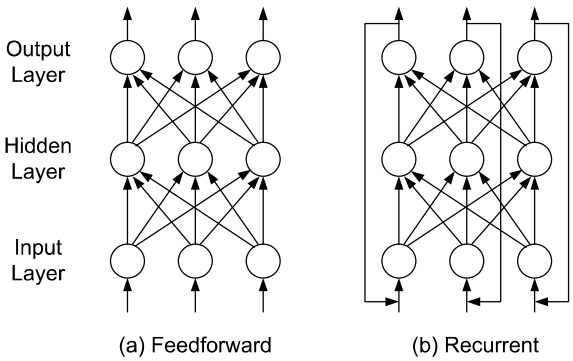

Figure 9.7 Neural network architectures

A recurrent neural network, on the other hand, distinguishes itself from a feed-forward network in that it has at least one feedback loop. For example, a recurrent neural network consists of a layer of neurons, some of which feed their outputs back as the inputs to neurons in the previous layer, as illustrated in Fig. 9.7(b). Owing to its particularly recurrent structure, the recurrent network can store information for future reference and thus is able to learn temporal as well as spatial patterns. This makes it useful in signal processing and prediction, where time plays a dominant role.

In Eq. 9.6, the average-flow-length of the nth sample, afl_n, depends on not only the information of the same sample, but also the last measurement (afl_{n-1}). The recurrent neural network, depicted in Fig. 9.8, is a suitable structure to model such a dynamic relation. Training of such a recurrent neural network, however, is difficult, as most existing training algorithms are derivative-based. For a feed-forward network, it is easy to obtain the error derivative with respect to each weight, but for a recurrent network, the training becomes complicated due to the interaction among the parameters of different times; as a result, the training tends to be trapped at local minimums. A feed-forward neural network structure is, therefore, recommended for the training of recurrent networks in many cases [212–216]. A feed-forward neural network shown in Fig. 9.9, consisting of two hidden layers with 15 neurons in the first layer and 20 in the second, is used in the training stage. It has been shown that a two-hidden-layer neural network can approximate any continuous nonlinear relations [213]. The number of neurons used is obtained based on a combination of experience and trial-and-error tests.

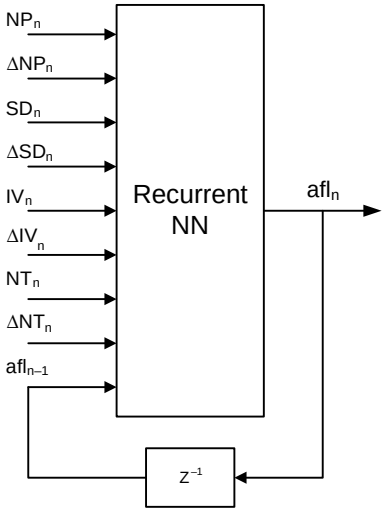

Figure 9.8 Structure of recurrent neural network model

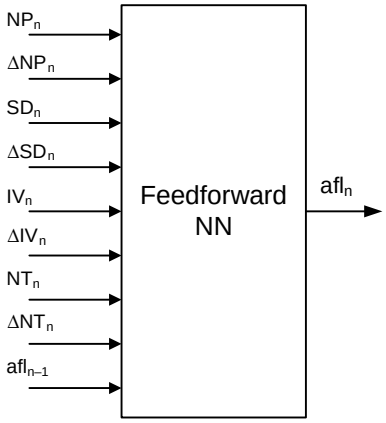

Figure 9.9 Structure of feed-forward neural network model

9.3.3 Training Algorithm

During the training stage, the weights and biases of the network were iteratively adjusted to minimize a defined network objective function, typically the mean squared error (MSE): the average squared error between the network outputs and the target outputs. A prescribed set of well-defined rules for the solution of the training problem is called a training algorithm. As one can expect, there are many algorithms that can be used for the network training, differing from each other in the way in which the weights are adjusted. The Levenberg-Marquardt (LM) algorithm, a training algorithm well known for its fast convergence speed and small residual training error [215–218], was employed in this work. The updating rule of the LM algorithm is represented as

$$\mathrm{D}W = \left(J^T J + \mathrm{m}I\right)^{-1} J^T e \tag{9.7}$$

where DW is the weight increment, J the Jacobian matrix of derivatives of each error with respect to each weight, I the identity matrix, m a scalar, and e the calculated MSE. When m is zero, the LM becomes a Gauss-Newton method using an approximate Hessian matrix; when m is large, the LM becomes the gradient descent method with a small step size. As the Gauss-Newton method is faster and more accurate in approaching the error minimum, m is adjusted in such a way to shift the LM toward the Gauss-Newton method as quickly as possible. The value of m is decreased after each successful step (reduction in MSE) and is increased only when a tentative step increases the error. In this way, it ensures a rapid reduction of the objective function. This method has a faster converging rate and a smaller residual training error compared with the commonly used back-propagation (BP) algorithm in network training [217].

9.3.4 Data Collection of Training and Validation Samples

The training and validation data were collected from experiments conducted on the reciprocating-screw injection molding machines using mold inserts with the capacitive transducer. The material used in the experiments was high-density polyethylene (HDPE) (Marlex, HMN6060).

Two sets of mold inserts were designed for the experiments. The first set, containing nine different inserts, are shown in Fig. 9.10. This set of inserts covers the basic mold shapes that the melt flow typically encounters, including constant area, gradually increasing/decreasing area, and abruptly increasing/decreasing area. A complex mold may be considered as a combination of those basic shapes. This set of mold inserts is referred to as Molds 1 to 9; they were used to generate the training data samples for the network.

Another set of molds with the same thickness but more complex geometry was also designed, as illustrated in Fig. 9.11, and referred to as Molds 10 and 11 for validation data generation. These molds combine several basic shapes of the first set. For example, the cross section of Mold 10 first decreases before it becomes a constant, similar to the shape of the first part of Mold 3; the middle section of Mold 10 is similar to Mold 9 before it is restored to the shape of the last part of Mold 3. Similarly, Mold 11 is geometrically similar to the combination of Molds 6 and 8. The design objective of this second set of molds is to demonstrate that the neural network model trained with data from the first simple set of molds can be used to predict the flow behavior for complex molds like Molds 10 and 11.

The injection velocity range for the machine was limited, due to its limited hydraulic power. Eleven different injection velocity profiles were designed to cover as wide as possible molding conditions for each mold, as illustrated in Fig. 9.12. They could be classified into three types: constant, step-change, and ramp profiles. There were five constant profiles, in which the injection velocity was set at 10 mm/s, 15 mm/s, 20 mm/s, 25 mm/s, and 30 mm/s, respectively. The step-change group consisted of two step-up and two step-down profiles, including 10 to 30 mm/s, 15 to 25 mm/s, 25 to 15 mm/s, and 30 to 10 mm/s. The step change was introduced at different times for different velocities and molds, but all around the time when the mold was half filled. For two ramp profiles, 10 to 30 mm/s ramp-up and 30 to 10 mm/s ramp-down, the ramp started at the point when the melt just entered the mold cavity; prior to that, the velocity was set at a constant rate for filling of sprue and runner. During filling, the sampling rate was set at 5 ms, which could guarantee that more than 150 samples could be collected, even for the fastest filling case. To avoid the local-minimum problem, different initial values were given in the network training; the results with the small errors were employed for prediction.

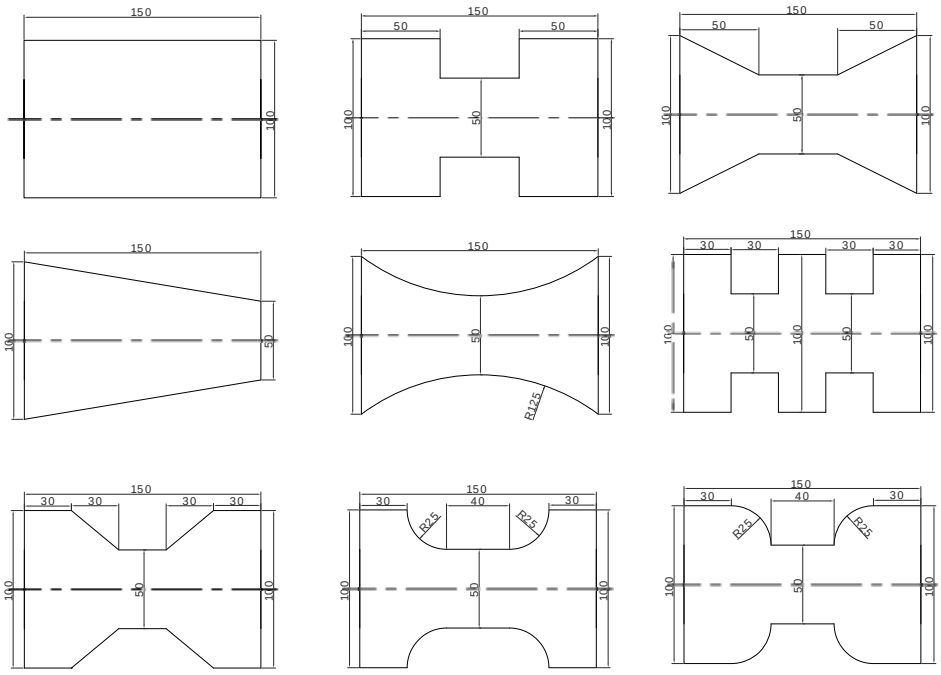

Figure 9.10 Geometry of the training mold inserts

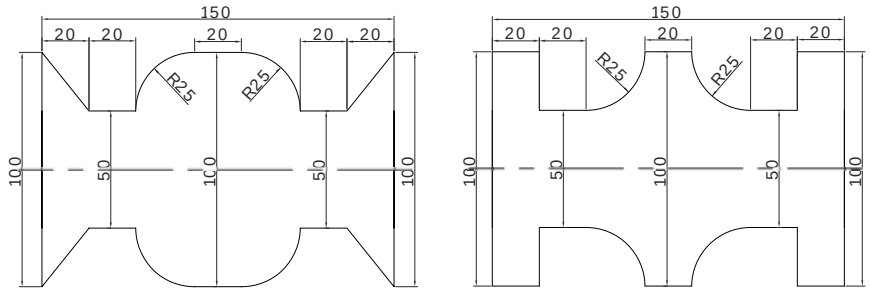

Figure 9.11 Geometry of validation molds

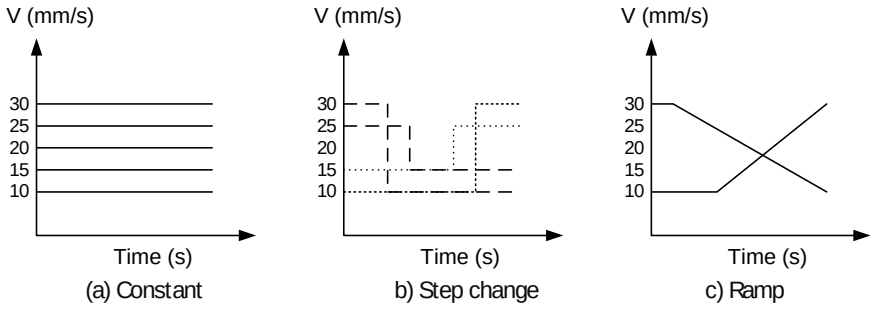

Figure 9.12 Injection velocity profiles

9.3.5 Model Performance

Different from the training stage, the recurrent neural network model structure should be used for the soft-sensor prediction because the preceding average-flow-length is not available. This results in an output error different from that of the training, even for the training samples. Because of this, small training errors may not necessarily guarantee a good approximation. Errors for both the training and validation samples are given in the following.

Figure 9.13 compares the predicted average-flow-length and the actual measurement of the capacitive transducer for Molds 1 to 9 with constant 20 mm/s injection velocity. With different mold geometry, the curve shapes of the average-flow-length are also different; they can be well predicted with a small error for all nine training molds.

The average-flow-length curve is influenced by not only the mold geometry, but also the injection velocity profile. Figure 9.14 presents the results with a step-change profile, in which the injection velocity changes from 30 mm/s to 10 mm/s during filling. Though the shapes are very different from those in Fig. 9.13, good agreement between network predictions and actual data can be obtained. This suggests that the neural network model has been well trained to predict the velocity change.

Experimental data from Molds 10 and 11 with 11 different injection velocity profiles are shown in Figs. 9.15 and 9.16, respectively. Although a slightly larger discrepancy exists between the predicted results and the CT measurements in comparison with those of the training samples, we still can see that the network outputs are in agreement with the true measurements in nearly all cases. The results of Mold 11 are better than those of Mold 10, possibly because Mold 11 is geometrically more similar to the training molds. If we define a geometry set to include the basic geometry elements appearing in the molds, we can see that both Mold 10 and Mold 11 are composed of seven geometry elements along the melt flow path. All seven geometry elements of Mold 11 have appeared in the training molds. Mold 10, however, is different from the training geometry set in the early converging and the last diverging sections. Though similar geometry elements are also present in Mold 3, the converge and diverge rates of the cross section in Mold 10 are larger. This suggests that the performance of the developed neural network model can be improved for prediction of the average-flow-length if the geometry set for training can be further enriched.

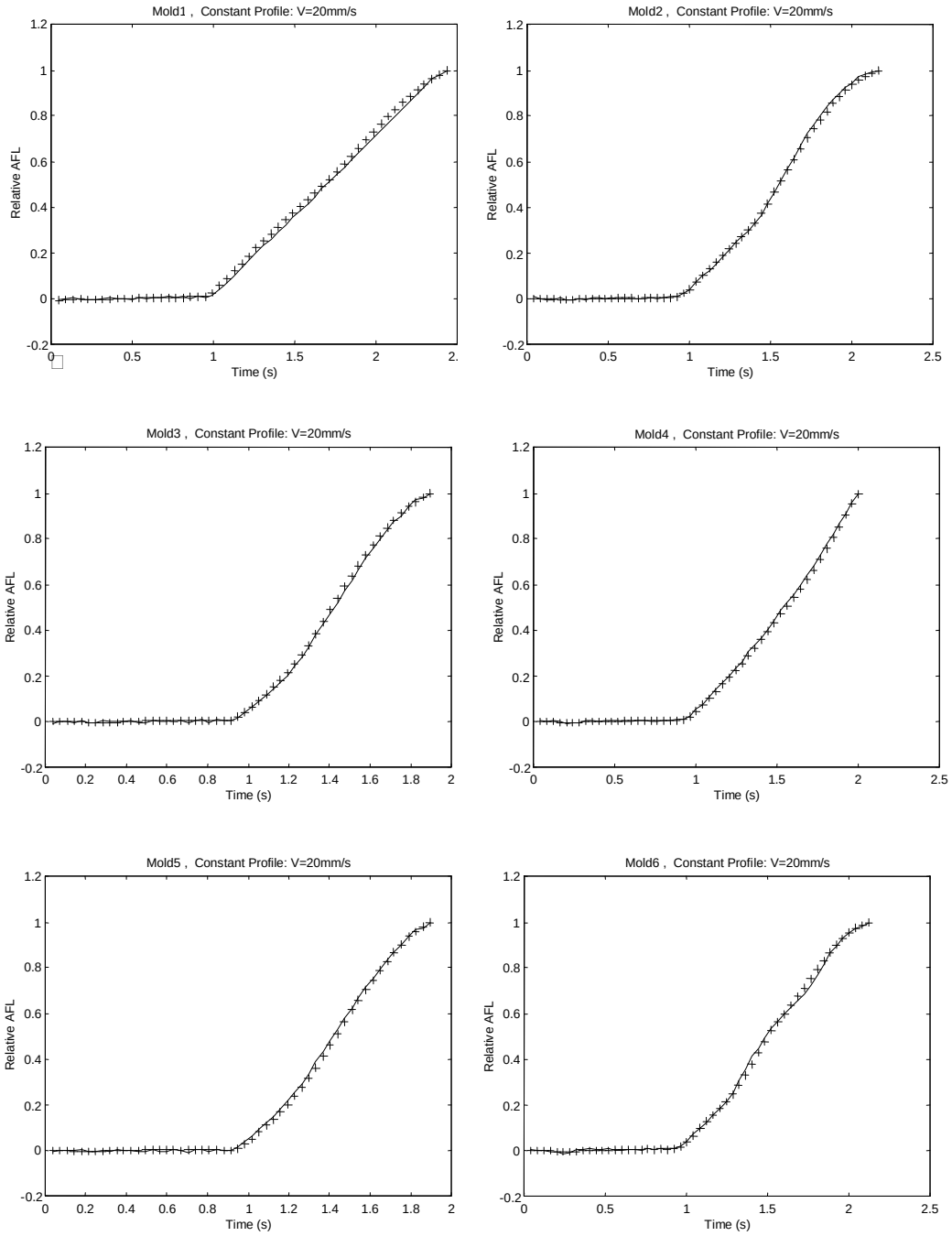

Figure 9.13 Results of different molds under a constant 20 mm/s profile (+: network output; –: CT measurement)

in Eq. 10.13, represents the ratio of relative volume change over relative pressure change. It relates to the material compression capability. Wide ranges are selected for these two coefficients to allow wide application conditions. In the tests, e_a is set as 30% and e_b 20%.

A simplified flowchart of the online detection system is illustrated in Fig. 10.3. Different detection algorithms and paths are taken for the different packing cases. The system first determines if the packing pressure is constant. This can be prior known information, or it can be determined by the detection system by comparing the pressure measurements. Packing ceases when the corresponding conditions are met.

10.1.3 Tests of Constant Packing Pressure Cases

Most existing machines have the packing pressure set at a constant during packing-holding. The detection of the gate freezing-off point with a constant packing profile is first conducted. A box mold with a spider gate as shown in Fig. 10.4 is used for the tests. The tests are conducted with different materials, including high-density polyethylene (HDPE), low-density polyethylene (LDPE), polypropylene (PP), and polystyrene (PS). In the following, unless otherwise specified, the results are obtained with the box mold, HDPE, and a 200°C nozzle heater temperature setting. In all experiments, the screw rotation speed during plastication is set at 80 RPM, the injection velocity 30 mm/s, and the sampling period of packing 15 ms. The end of packing is determined online by the detection system, and the results are compared with the established offline measurements.

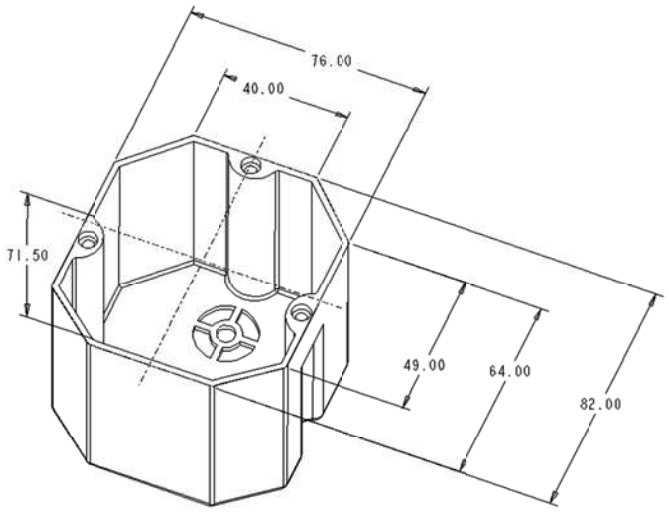

Figure 10.4 Geometry of box mold

Gate Freezing Detection for a Constant Packing Pressure

The first test is conducted with a closed-loop packing-holding control at a 300 bar constant pressure. The threshold representing *SMALL* in Rule 3 is set at 10%. The gate freezing-off point is detected to be 10.7 s by the autodetection system, according to Rule 3. The screw displacement is shown in Fig. 10.5, in which both the raw and filtered data are illustrated. A small delay is introduced by the filter, but it does not affect the detection result significantly. In addition, to avoid the possible pressure variation around the V/P transfer, data for the first half-second of packing are not used in the detection. As previously analyzed, the figure indicates that the screw displacement levels off just before packing ends. The corresponding normalized standard deviation, defined in Eq. 10.17, is illustrated in Fig. 10.6, in which the values larger than one are truncated for better illustration. As suggested by the proposed Rule 3, a rapid decrease of the defined standard deviation can be observed just before the packing stops.

To evaluate the developed detection system, a number of moldings are made with the same pressure setting but different packing times. Their part weights are measured and compared with that of the autodetection system, as listed in Table 10.1. It can be seen that the part weight increases with the packing time, and the increase in packing time from 10 s to 14 s is small. The part weight of the autodetection system is 47.78 g, the same as that packed for 14 s. This suggests that the gate has already frozen when packing beyond the time suggested by the online detection system. On the other hand, a significant weight difference between that of the autodetection and those packed under 10 s indicates that gate has not frozen before 10 s of packing.

In the detection system, a threshold representing *SMALL* for the normalized standard deviation used in Rule 3 was set at 10%. Experiments have also been conducted to test the sensitivity of the detection system with different thresholds. The results show that the system is not very sensitive to the threshold, as the part weight variation is small. The best threshold can be determined by finding the minimal packing time but with a maximum weight. This system has also been successfully tested with different packing pressures and different molds. Detailed results can also be found in reference [226].

Table 10.1 Part Weight Comparison of 300 bar Constant Packing Pressure

	Results with fixed packing time					Results of detection system
Packing time (s)	3	6	9	10	14	10.7
Part weight (g)	45.85	46.90	47.60	47.72	47.78	47.78

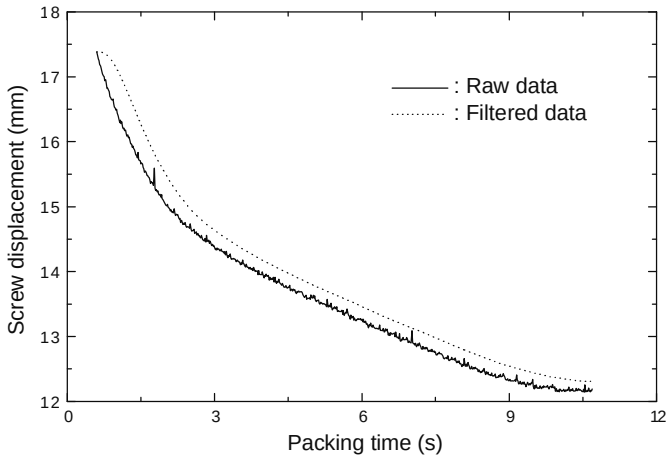

Figure 10.5 Screw displacement with 300 bar constant packing profile

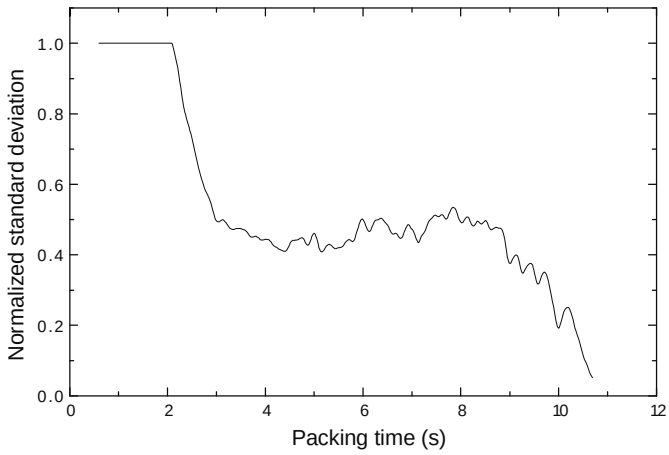

Figure 10.6 Plot of normalized standard deviation

Online Detection Results during Warm-Up Period

In practice, the machine takes a number of cycles to reach the steady state after it starts. This is referred to as the warm-up period. Typically, the part weight varies during this period if it is packed with a fixed length. The gate freezing time during this transient period may vary cycle to cycle. The autodetection system can compensate for the warm-up by automatically varying the packing time to keep a relatively constant part weight.

Experiments are conducted for a series of cycles immediately after the machine starts. The autodetected gate freezing-off times, together with the corresponding part weights, are shown in Fig. 10.7.

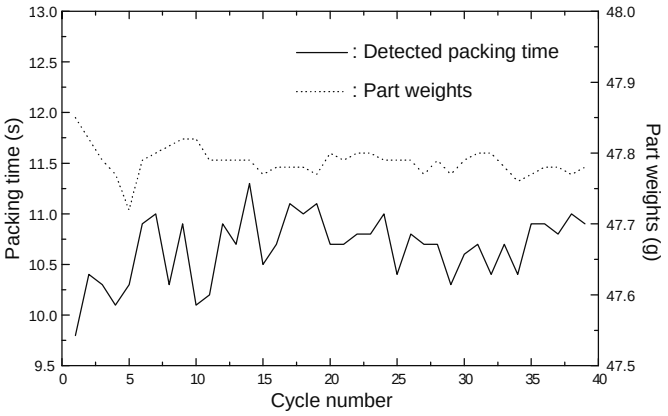

Figure 10.7 Variations of packing time and part weights cycle to cycle

As the mold coolant temperature is controlled by a cooling system, the cycle variations are not large. At the very beginning of the molding, an increasing trend for packing time and a decreasing trend for part weight can be observed. The increase in the gate freezing-off time is due to fact that the gate becomes warmer from cycle to cycle, which in turn slows down the gate freezing; the decrease in the part weight is a result of the increasing mold temperature, which consequently reduces the part shrinkage. After several cycles, the part weight obviously becomes steady, with a variation less than 0.1%.

The packing times and part weights of the last 10 cycles of Fig. 10.7 are listed in Table 10.2. The detected packing time varies from 10.4 s to 11.1 s, but the part weights are very close, with a maximum relative variation less than 0.084%, indicating good performance of the developed system.

Table 10.2 Online Detection Results of Different Cycles

Cycle number	Detected packing time (s)	Part weights (g)
30	10.6	47.79
31	10.7	47.80
32	10.4	47.80
33	10.7	47.78
34	10.4	47.76
35	10.9	47.77
36	10.9	47.78
37	10.8	47.78
38	11.1	47.77
39	10.9	47.78

Subject Index