Computers and Chemical Engineering 93 (2016) 284–292

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

Iterative learning model predictive control for constrained multivariable control of batch processes

Se-Kyu Oh, Jong Min Lee*

School of Chemical and Biological Engineering, Institute of Chemical Processes, Seoul National University, 1 Gwanak-ro, Gwanak-gu, Seoul 08826, Republic of Korea

ARTICLE INFO

Article history:
Received 18 December 2015
Received in revised form 16 May 2016
Accepted 3 July 2016
Available online 5 July 2016

Keywords:
Iterative learning control
Model predictive control
Disturbance rejection
Offset-free control
Constrained multivariable control
Iterative learning model predictive control

ABSTRACT

In this paper, we propose a model predictive control (MPC) technique combined with iterative learning control (ILC), called the iterative learning model predictive control (ILMPC), for constrained multivariable control of batch processes. Although the general ILC makes the outputs converge to reference trajectories under model uncertainty, it uses open-loop control within a batch; thus, it cannot reject real-time disturbances. The MPC algorithm shows identical performance for all batches, and it highly depends on model quality because it does not use previous batch information. We integrate the advantages of the two algorithms. The proposed ILMPC formulation is based on general MPC and incorporates an iterative learning function into MPC. Thus, it is easy to handle various issues for which the general MPC is suitable, such as constraints, time-varying systems, disturbances, and stochastic characteristics. Simulation examples are provided to show the effectiveness of the proposed ILMPC.

1. Introduction

Iterative learning control (ILC) is an effective control technique for improving the tracking performance of a batch process under model uncertainty. ILC was originally introduced for robot manipulators (Arimoto et al., 1984) and has been implemented in many industrial processes, such as semiconductor manufacturing and chemical batch processes (Xu et al., 1999; Ahn et al., 2014). In many ILC algorithms, the input sequences for the current batch are calculated using the tracking error sequences of the previous batch. This type of ILC algorithm uses open-loop control within a batch and cannot handle real-time disturbances. ILC should be integrated with real-time feedback control to reject real-time disturbances.

Model predictive control (MPC) has become the accepted standard for complex constrained multivariable control problems in the process industry. Some studies about ILC formulations combined with MPC, called iterative learning model predictive control (ILMPC), have been studied for handling real-time disturbances in batch processes. In case of combining ILC with MPC, it should include fundamental advantages of MPC as well as real-time feedback function. The following characteristics of MPC should be included in ILMPC. (1) ILMPC should guarantee offset-free control. (2) It should have a single optimization step, not two optimization steps for both ILC part and MPC part separately. (3) It should consider constraints and ensure that a feasible solution will always be found. (4) Prediction horizon should be able to be adjusted to reduce the computational load. (5) If the model is linear time-invariant (LTI), it should use LTI model directly. However, there are no studies about ILMPC algorithms which contain above all advantages. Most studies of ILMPC use a state-space model where a state vector consists of the entire error sequences of a batch, and a prediction horizon is fixed as the entire batch horizon (Lee et al., 2000; Chin et al., 2004; Xiong et al., 2005; Liu and Kong, 2013). Thus, the control calculations may not be performed within a sampling interval if a process has output constraints, a small sample time, long operation time, and many outputs. The prediction horizon should be adjusted to reduce the excessive computational load. In addition, the resulting formulation of such an algorithm employs linear time-varying (LTV) models in the state augmentation step, even if a system is LTI. Thus, additional calculation burdens for LTI systems are added in the algorithm. Two-stage approaches have also been proposed for combining ILC with real-time feedback (Chin et al., 2004; Xiong et al., 2005; Lu et al., 2015). The two-stage approaches have difficulties in system analysis and parameter tuning and require two optimization steps. In addition, the time-wise feedback controllers of the approaches are not offset-free control; thus, offset occurs in the early batches until the batch-wise controller shows convergence. Above all, the two-stage approaches do not consider

* Corresponding author.
E-mail address: jongmin@snu.ac.kr (J.M. Lee).

http://dx.doi.org/10.1016/j.compchemeng.2016.07.011
0098-1354/

constraints. ILC combined with dynamic matrix control (DMC) for LTI system (Mo et al., 2012) has been proposed, but the DMC algorithm without an observer cannot handle unknown disturbance and measurement noise effectively (Lundstrom et al., 1995).

In this paper, we proposed ILMPC which contains fundamental advantages of MPC. (1) The proposed ILMPC guarantees offset-free control by introducing incremental state-space model. Therefore, outputs can track reference trajectories while rejecting disturbances at the first batch even if this is not perfect tracking. (2) This is one-stage approach and has single optimization step. (3) It considers constraints and includes slack variable; thus this algorithm ensures that a feasible solution will always be found. (4) Prediction and control horizon can be adjusted to reduce the computational load. (5) If the system is LTI, this algorithm uses LTI parameters directly. Finally, the formulation and algorithm procedure of the proposed ILMPC is similar to conventional MPC; thus various techniques for MPC can be applied for the proposed ILMPC without particular modification. The reason which these advantages can be included in the proposed algorithm is that a prediction model formulated by an input–output model between two adjacent batches is directly applied to the algorithm in an identical way as a conventional MPC.

The rest of the paper is organized as follows. In Section 2, a prediction model for ILMPC is derived. In Section 3, unconstrained and constrained ILMPC controllers and the convergence property are provided. Numerical illustrations for unconstrained single-input single-output (SISO), constrained multiple-input multiple-output (MIMO) and nonlinear processes are provided in Section 4. Finally, concluding remarks are given in Section 5.

2. Prediction model for iterative learning model predictive control

2.1. Incremental state-space model

First, we consider the following linear discrete time-invariant system which operates on an interval $t \in [0, N]$:

$$\begin{aligned} x_k(t+1) &= Ax_k(t) + Bu_k(t) \\ y_k(t) &= Cx_k(t) \end{aligned} \tag{1}$$

where $x_k(t) \in \mathbb{R}^{n_x}$ is the state vector; $u_k(t) \in \mathbb{R}^{n_u}$ is the input vector; $y_k(t) \in \mathbb{R}^{n_y}$ is the output vector; t is the time index; k is the batch index; and the matrices A, B, and C are real matrices of appropriate dimensions. An incremental state-space model uses the control increment instead of the control signal. This model can be written in the general state-space form with $\delta u_k(t) = u_k(t) - u_k(t-1)$. The following representation is the augmented incremental state-space model:

$$\begin{aligned} \begin{bmatrix} x_k(t+1) \\ u_k(t) \end{bmatrix} &= \overbrace{\begin{bmatrix} A & B \\ 0 & I \end{bmatrix}}^{\tilde{A}} \overbrace{\begin{bmatrix} x_k(t) \\ u_k(t-1) \end{bmatrix}}^{\tilde{x}_k(t)} + \overbrace{\begin{bmatrix} B \\ I \end{bmatrix}}^{\tilde{B}} \delta u_k(t) \\ y_k(t) &= \underbrace{\begin{bmatrix} C & 0 \end{bmatrix}}_{\tilde{C}} \begin{bmatrix} x_k(t) \\ u_k(t-1) \end{bmatrix} \end{aligned} \tag{2}$$

The characteristic polynomial equation of the augmented model is

$$\rho(\lambda) = \det \begin{bmatrix} \lambda I - A & -B \\ 0 & I - \lambda I \end{bmatrix} = (\lambda - 1)^{n_u} \det(\lambda I - A) = 0 \tag{3}$$

This means that there are n_u integrators are embedded in the augmented model. Defining a new state vector as $\tilde{x}_k(t) \triangleq \begin{bmatrix} x_k(t)^T & u_k(t-1)^T \end{bmatrix}^T$, the incremental model takes the following general form:

$$\begin{aligned} \tilde{x}_k(t+1) &= \tilde{A}\tilde{x}_k(t) + \tilde{B}\delta u_k(t) \\ y_k &= \tilde{C}\tilde{x}_k \end{aligned} \tag{4}$$

It is called an incremental state-space model (Camacho and Alba, 2013) or a state-space model with embedded integrator (Wang, 2009).

2.2. Prediction model

The system (4) can be rewritten as a lifted system because finite time intervals $[0, N]$ are considered in ILMPC:

$$\hat{\mathbf{y}}_k = \mathbf{G}_m \delta \mathbf{u}_k + \mathbf{F}_m \tilde{x}_k(0) \tag{5}$$

where $\mathbf{G}_m \in \mathbb{R}^{(n_y N) \times (n_u N)}$ and $\mathbf{F}_m \in \mathbb{R}^{(n_y N) \times n_x}$ are defined as

$$\mathbf{G}_m \triangleq \begin{bmatrix} \tilde{C}\tilde{B} & 0 & \cdots & 0 \\ \tilde{C}\tilde{A}\tilde{B} & \tilde{C}\tilde{B} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{C}\tilde{A}^{N-1}\tilde{B} & \tilde{C}\tilde{A}^{N-2}\tilde{B} & \cdots & \tilde{C}\tilde{B} \end{bmatrix}, \quad \mathbf{F}_m \triangleq \begin{bmatrix} \tilde{C}\tilde{A} \\ \tilde{C}\tilde{A}^2 \\ \vdots \\ \tilde{C}\tilde{A}^N \end{bmatrix} \tag{6}$$

and the vectors $\hat{\mathbf{y}}_k \in \mathbb{R}^{n_y N}$ and $\mathbf{u}_k \in \mathbb{R}^{n_u N}$ are defined as

$$\hat{\mathbf{y}}_k \triangleq \begin{bmatrix} \hat{y}_k(1)^T & \hat{y}_k(2)^T & \cdots & \hat{y}_k(N)^T \end{bmatrix}^T \tag{7}$$

$$\delta \mathbf{u}_k \triangleq \begin{bmatrix} \delta u_k(0)^T & \delta u_k(1)^T & \cdots & \delta u_k(N-1)^T \end{bmatrix}^T \tag{8}$$

The input–output relationship between two adjacent batches is

$$\hat{\mathbf{y}}_k = \mathbf{y}_{k-1} + \mathbf{G}_m \Delta \delta \mathbf{u}_k + \mathbf{F}_m \Delta \tilde{x}_k(0) \tag{9}$$

where ˆ(hat) means predicted value, δ is a time-increment operator and Δ is a batch-increment operator. That is, $\Delta\delta u_k(t) = \{u_k(t) - u_k(t-1)\} - \{u_{k-1}(t) - u_{k-1}(t-1)\}$ and $\Delta\tilde{x}_k(0) = \tilde{x}_k(0) - \tilde{x}_{k-1}(0)$. Then, the following representation can be obtained using $\mathbf{e}_k = \mathbf{r} - \mathbf{y}_k$ where $\mathbf{r}$ is the reference trajectory.

$$\hat{\mathbf{e}}_k = \mathbf{e}_{k-1} - \mathbf{G}_m \Delta \delta \mathbf{u}_k - \mathbf{F}_m \Delta \tilde{x}_k(0) \tag{10}$$

The basic assumption of ILC is an identical initialization condition ($\tilde{x}_k(0) = \tilde{x}_{k-1}(0)$); thus, $\Delta\tilde{x}_k(0)$ is zero (Arimoto et al., 1984; Moore, 2012). However, we do not remove the initial state term because it is used for deriving free response term including estimated current states which are necessary for real-time feedback. At time t of the kth batch, future predictions up to a prediction