Receding Horizon Control for High-Dimensional Burgers’ Equations with Boundary Control Inputs

By Tomoaki Hashimoto,1 Yu Takiguchi2 and Toshiyuki Ohtsuka1

1Department of Systems Innovation, Osaka University, Toyonaka, Japan
2Horiba Advanced Techno Corporation, Kyoto, Japan

(Received June 19th, 2012)

Receding horizon control is a feedback control approach that optimizes control performance over a finite horizon, and its performance index has moving initial and terminal times. Controlling the flow of fluids is a challenging problem that arises in many fields including aeronautical, biological and chemical engineering. The objective of this study is to provide a novel framework for designing a receding horizon controller for high-dimensional Burgers’ equations used to describe fundamental flow phenomena. The advantage of our proposed method is that it can be applied to a wide class of optimization problems of high-dimensional Burgers’ equations. The effectiveness of the proposed method is verified by numerical simulation.

Key Words: Non-linear Systems, Optimal Control, Fluid Dynamics, Numerical Solution

Nomenclature

\[ \mathbb{R} \]: real numbers
\[ \mathbb{R}_+ \]: non-negative real numbers
\[ \mathbb{N}_+ \]: positive integers
\[ x \in \mathbb{R}^n \]: spatial vector
\[ t \in \mathbb{R}_+ \]: temporal variable
\[ z \in \mathbb{R}^n \]: flow velocity
\[ v \in \mathbb{R} \]: viscosity coefficient
\[ u \in \mathbb{R}^n \]: control input

1. Introduction

Controlling fluid dynamics is a challenging problem that arises in the field of aerospace engineering. It is well known that fluid flow is governed by Navier-Stokes equations. Burgers’ equations are also known as the simplest partial differential equations (PDEs) that can be used to describe various flow phenomena. We can obtain Burgers’ equations by eliminating the pressure term in Navier-Stokes equations. Burgers’ equations consist of the advective and diffusive terms, which can be used to represent fundamental properties of flow phenomena. Therefore, using Burgers’ equations can be regarded as a natural first step towards developing a method for controlling flows.

The boundary feedback control problem for a class of systems described by Burgers’ equations was investigated in Refs. 1)–7). A boundary control law that uses linearization and achieves local stabilization of Burgers’ equations was proposed in Ref. 1). However, this control law requires the initial solution to be sufficiently small. By removing this restriction on the size of the initial solution, the global existence and uniqueness of a solution of a Burgers’ equation were shown in Ref. 2). The control methods proposed in Ref. 2) are based on a local Lyapunov function, hence the initial states of a system should be given within a local attractor. To overcome this local-stability restriction, the global exponential stability of Burgers’ equations in the \( L^2 \) and \( H^1 \) norms was investigated in Refs. 3) and 4), respectively. Furthermore, a backstepping boundary control law applying actuator dynamics was proposed for Burgers’ equations in Ref. 5). For practical applications in which the viscosity is unknown, an adaptive control of Burgers’ equations was proposed in Ref. 6). Moreover, the adjoint-based optimal control method7) was proposed for Burgers’ equations using the high-order spectral difference method.

Although the aforementioned studies1–7) have achieved tremendous progress in controlling the one-dimensional Burgers’ equation, the control problem of higher-dimensional Burgers’ equations still remains unsolved. Furthermore, because the control design methods presented in Refs. 1)–7) utilize a variable transformation, which reduces the Burgers’ equation into a simple diffusion equation, it is difficult to extend them to higher-dimensional systems. In general, it is difficult to find such transformations for high-dimensional systems.

Receding horizon control is a feedback control scheme, in which its performance index has moving initial and terminal times. An efficient algorithm, called C/GMRES,9) was proposed for solving receding horizon control problems for non-linear systems described by ordinary differential equations. However, suitable reformulation and modification are necessary to apply the C/GMRES method to systems described by PDEs. Recently, we proposed a methodology for designing receding horizon controllers for a particular class of one-dimensional non-linear PDEs9,10). It was shown in Ref. 9) that the C/GMRES algorithm can be applied to solve the obtained optimality conditions for non-linear PDEs. Motivated by the fact that the obtained stationary
conditions for the optimization problem of nonlinear PDEs have a particular structure with respect to unknown parameters, we developed an efficient algorithm called the contraction mapping method, instead of using the CGMRES algorithm for numerically solving the stationary conditions. However, the methods proposed in Refs. 9 and 10 cannot be applied to high-dimensional PDEs. Hence, the objective of this study is to propose a method for designing receding horizon controllers for high-dimensional Burgers’ equations with constrained states and inputs. An advantage of receding horizon control is that we can address both the state variable and control input constraints. The method proposed here is advantageous for its applicability to a wide class of optimization problem, for high-dimensional Burgers’ equations.

This paper is organized as follows. In section 2, we introduce some notations and the system model. In section 3, we consider the receding horizon control problem for Burgers’ equations subject to constraints. Moreover, using the variational principle, we derive the stationary conditions, which must be satisfied for optimizing the performance index. In section 4, we provide a brief description of the algorithms used to numerically solve the stationary conditions. In section 5, we provide an illustrative example that verifies the effectiveness of the proposed method. Finally, we provide concluding remarks in section 6.

2. Notations and System Model

The transpose of matrix $A$ is denoted by $A'$. Let $\text{diag}\{\cdots\}$ denote a diagonal matrix. Furthermore, let $z = [z_1, \cdots, z_n]'$ and $x = [x_1, \cdots, x_n]'$ denote the state and spatial vectors, respectively. Let $t$ denote the temporal variable. Without loss of generality, we restrict our attention to the range $0 \leq x_i \leq h$ for all $i = 1, \cdots, n$, where $h$ is a positive constant. Let $\Omega$ and $\partial \Omega$, be sets defined by

$$\Omega := \bigcup_{i=1}^{n} \{x_i|0 \leq x_i \leq h\}$$

and

$$\partial \Omega := \{x_i|x_i = 0, x_i = h\} \cap \Omega,$$

respectively. Let $z(x, t) : \Omega \times \mathbb{R}_+ \rightarrow \mathbb{R}^n$ be a continuous vector-valued function with respect to $x$ and $t$. Then, for $i = 1, \cdots, n$ and $j = 1, 2$, we introduce the following notations.

$$\dot{z}_i(x, t) := \frac{\partial z_i(x, t)}{\partial t} := \left[\frac{\partial z_1(x, t)}{\partial t}, \cdots, \frac{\partial z_n(x, t)}{\partial t}\right]'$$

$$z_{x_i}(x, t) := \frac{\partial z_i(x, t)}{\partial x_i} := \left[\frac{\partial^2 z_1(x, t)}{\partial x_1^2}, \cdots, \frac{\partial^2 z_n(x, t)}{\partial x_n^2}\right]'$$

$$z_{x_i x_j}(x, t) := \frac{\partial^2 z_i(x, t)}{\partial x_i \partial x_j} := \left[\frac{\partial^2 z_1(x, t)}{\partial x_1^2}, \cdots, \frac{\partial^2 z_n(x, t)}{\partial x_n^2}\right]'$$

$$\nabla := \left[\frac{\partial}{\partial x_1}, \cdots, \frac{\partial}{\partial x_n}\right]', \quad \nabla^2 := \frac{\partial^2}{\partial x_1^2} + \cdots + \frac{\partial^2}{\partial x_n^2}$$

$$z \cdot \nabla := z_1 \frac{\partial}{\partial x_1} + \cdots + z_n \frac{\partial}{\partial x_n}.$$
with the Burgers’ equation, Eq. (3), and the equality constraint, Eq. (4), respectively.

\[ \dot{J} = \int_{\Omega} \phi(z(x, t + T)) \, dx + \int_t^{t+T} \int_{\Omega} \left( H - \lambda'(x, \tau) \frac{\partial z(x, \tau)}{\partial \tau} \right) \, dx \, d\tau. \]  

(6)

\[ \int_t^{t+T} \int_{\Omega} -\lambda'(x, \tau) \frac{\partial \delta z(x, \tau)}{\partial \tau} \, dx \, d\tau = \left[ \int_{\Omega} -\lambda'(x, \tau) \delta z(x, \tau) \, dx \right]_t^{t+T} + \int_t^{t+T} \int_{\Omega} \left( \frac{\partial \lambda(x, \tau)}{\partial \tau} \right)' \delta z(x, \tau) \, dx \, d\tau \]

(8)

In Eq. (8), we set \( \delta z(x, \tau) = 0 \) because \( z(x, \tau) \) is fixed at \( \tau = t \) as the present state. Using Eq. (8), \( \delta z_{\tau} \) can be converted into \( \delta z \). Furthermore, the following integration by parts can be used to compute \( \delta J \).

[Equation content continued]

\[ \int_{\Omega} \frac{\partial H}{\partial z_{\tau}} \delta z_{\tau} \, dx = \int_{\Omega} \frac{\partial H}{\partial z} \delta z \, dx - \int_{\Omega} \frac{\partial}{\partial \tau} \left( \frac{\partial H}{\partial z} \right) \delta z \, dx \]

(9)

\[ \int_{\Omega} \frac{\partial H}{\partial z} \delta z \, dx = \int_{\Omega} \left[ \frac{\partial H}{\partial z} G_i(x) \delta u_i \right]_0^h - \left[ \frac{\partial}{\partial \tau} \left( \frac{\partial H}{\partial z} \right) \delta z \right]_0^h \, dx + \int_{\Omega} \frac{\partial}{\partial x_i} \left( \frac{\partial H}{\partial z} \delta z \right) \, dx \]

(10)

From the boundary condition, Eq. (2), it follows that

\[ \delta z_{\tau}(x, t) = G_i(x) \delta u_i(x, t). \]

(11)

By substituting Eq. (11) into Eq. (10) for \( x \in \partial \Omega_i \), we obtain

\[ \int_{\Omega_i} \frac{\partial H}{\partial z} \delta z \, dx = \int_{\Omega} \left[ \delta x_{\tau}(x, t) \delta z \, dx \right]_{\partial \Omega_i} + \int_{\Omega_i} \frac{\partial}{\partial \tau} \left( \frac{\partial H}{\partial z} \delta z \right) \, dx \]

(12)

Next, we apply Eqs. (8), (9) and (12) to compute \( \delta J \) as follows.

\[ \delta J = \int_{\Omega} \left( \frac{\partial \phi[z(x, t + T)]}{\partial z} - \lambda'(x, t + T) \right) \delta z(x, t + T) \, dx \]

\[ + \int_t^{t+T} \left[ \int_{\Omega} \delta \lambda' \left( A - \frac{\partial}{\partial \tau} \right) + \delta \mu' C \left[ \left( \frac{\partial \lambda}{\partial z} \right)' + \frac{\partial H}{\partial z} + \sum_{i=1}^n \sum_{j=1}^{2^{i-1}} (\delta z_{\tau}) \frac{\partial H}{\partial z_{\tau}} \right] \delta z \right] \, dx \]

\[ + \sum_{i=1}^n \int_{\Omega_i} \sum_{x_j \in [0, h]} \left( s(x_j) \left( \sum_{j=1}^2 (-1)^{j-1} \frac{\partial}{\partial x_j} \left( \frac{\partial H}{\partial z_{\tau}} \right) \delta z \right) \right) G_i \right] \, dx \]

(13)

On the basis of the variational principle, we obtain the necessary conditions for a stationary value of \( J \) over the horizon \( t \leq \tau \leq t + T \) as follows. For \( x \in \Omega \), we have

\[ \frac{\partial z}{\partial \tau} = A \left( z, \cdots, z_{\tau}, \cdots \right) \]  

(14a)

\[ \lambda(x, t + T) = \lambda \left( \frac{\partial \phi[z(x, t + T)]}{\partial z} \right)' \]  

(14b)

\[ \left( \frac{\partial \lambda(x, \tau)}{\partial z} \right)' = -\frac{\partial H}{\partial z} + \sum_{j=1}^{2^{i-1}} (\delta z_{\tau}) \frac{\partial H}{\partial z_{\tau}} \]  

(14c)

\[ C(u_i, z, \cdots, z_{\tau}, \cdots) = 0 \]  

(14d)

and for \( x \in \partial \Omega_i \) and \( i = 1, \ldots, n \), we have

\[ \sum_{j=1}^2 (-1)^{j-1} \frac{\partial}{\partial x_j} \left( \frac{\partial H}{\partial z_{\tau}} \right) = 0 \]  

(14e)

To minimize or maximize performance index of Eq. (6), conditions in Eqs. (14a)–(14f), termed stationary conditions, must be satisfied. A well-known difficulty in solving non-linear optimal control problems is that the obtained stationary conditions cannot be solved analytically in general. Therefore, several algorithms have been developed for numerically solving stationary conditions.

4. Numerical Solution

Although we have analytically derived the exact stationary conditions in section 3, we need a numerical algorithm for solving the stationary conditions. In the following, we...
provide a framework in which a fast on-line algorithm called C/GMRES\(^8\) is applicable for solving the receding horizon control problem in non-linear PDEs.

From the stationary conditions derived in section 3, Eqs. (14a) and (14c) are time-evolutionary equations with respect to \(z\) and \(\lambda\), while the remaining ones are algebraic equations. Moreover, the condition of Eq. (14e) is the boundary condition for the time-evolutionary equation of \(\lambda\). Let \(U\) be defined by \(U := [u_1, \ldots, u_n, \mu]^{T}\). Given an initial solution \(U(x, t)\), where \(t \in [t, t + T]\), then the present state is \(z(x, t)\). First, we compute \(z(x, t)\) for \(t \in [t, t + T]\) by numerically solving Eq. (14a) from \(t = t\) to \(t = t + T\), while satisfying the boundary condition of Eq. (2). Then, using the value of the terminal state \(z(x, t + T)\), we can apply Eq. (14b) and obtain the terminal costate \(\lambda(x, t + T)\). Consequently, \(\lambda(x, t)\) for \(t \in [t, t + T]\) can be computed by numerically solving Eq. (14c) from \(t = t + T\) to \(t = t\), while satisfying the boundary condition of Eq. (14e). Figure 1 illustrates the flow chart for solving the time-evolutionary equation of \(z(x, t)\) forward, while that for solving the time-evolutionary equation of \(\lambda(x, t)\) backward. Figure 2 illustrates the flow chart for solving the stationary conditions. For given \(U\) and \(z(x, t)\), \(z(x, t)\) and \(\lambda(x, t)\) for \(t \in [t, t + T]\) can be determined so as to satisfy Eqs. (14a)–(14c) and (14e). However, the remaining conditions in Eqs. (14d) and (14f) are not necessarily satisfied for given \(U\) and \(z(x, t)\). Therefore, \(U\) must be updated so as to satisfy Eqs. (14d) and (14f). The method of updating \(U\) is discussed later.

To solve the stationary conditions in Eq. (14) using a numerical algorithm, we must first discretize them into finite difference equations. Let \(x \in \Omega\) be divided into \(n_x\) grid points, and let \(\hat{x} := [\hat{x}_1, \ldots, \hat{x}_{n_x}] \in \mathbb{R}^{n_x}\) denote the discretized spatial vector. All discretized variables of each \(x_1, \ldots, x_n\) are unified into \(\hat{x}\). Likewise, let time \(t \in [t, t + T]\) over the prediction horizon be divided into \(n_t\) steps, and let \(\hat{t} := [\hat{t}_1, \ldots, \hat{t}_{n_t}] \in \mathbb{R}^{n_t}\) denote the discretized temporal vector. Note that \(\hat{t}_1\) is equal to the current time \(t\). Let the set \(\{\hat{\alpha}_{k,1}, \ldots, \hat{\alpha}_{k,n}\}\) be given by \(\{\hat{x}_1, \ldots, \hat{x}_{n_x}\} \cap \partial \Omega\). Let \(\hat{\alpha}_{k} \in \mathbb{R}^{n_x}\) be defined by \(\hat{\alpha}_{k} := [\hat{\alpha}_{k,1}, \ldots, \hat{\alpha}_{k,n}]\). Let \(\hat{A}(\hat{\alpha}_k, \hat{\beta}) := [\hat{A}_1(\hat{\alpha}_k, \hat{\beta}_1), \ldots, \hat{A}_n(\hat{\alpha}_k, \hat{\beta}_n)]\) denote the discretized control input. Let \(\hat{z}(\hat{x}, \hat{t})\), \(\hat{\lambda}(\hat{x}, \hat{t})\) and \(\hat{\mu}(\hat{x}, \hat{t})\) denote the discretized state, costate and Lagrange multiplier, respectively. Note that \(z(x, t)\) is forward, while that for solving the time-evolutionary equation of \(\lambda\) is backward, with \(t \in [t, t + T]\), where \(z(x, t)\) is calculated iteratively from \(k = 1, \ldots, n_t\). Note that \(z_k \in \mathbb{R}^{n_x}\) is equal to the current known state \(\hat{z}(\hat{x}, \hat{t})\). For other variables, we adopt a similar notation. The finite difference approximation results in the following discretized stationary conditions over the horizon \(k = 1, \ldots, n_t\).

\[
\begin{align*}
\hat{z}_{k+1} &= \hat{A}(z_k, u_k) \quad (15a) \\
\hat{\lambda}_n &= \hat{\Phi}(z_n) \\
\hat{\lambda}_k &= \hat{D}(z_{k+1}, z_k, u_{k+1}, \mu_{k+1}) \quad (15c) \\
\hat{C}(z_k, \hat{z}_k) &= 0 \quad (15d) \\
\hat{E}(\hat{u}_k, \hat{\mu}_k, \hat{z}_k, \hat{\lambda}_k) &= 0. \quad (15e)
\end{align*}
\]

Here, \(\hat{A}, \hat{\Phi}, \hat{D} \in \mathbb{R}^{n_x \times n_x}, \hat{C} \in \mathbb{R}^{n_x}\) and \(\hat{E} \in \mathbb{R}^{n_x}\) denote appropriately given vector-valued functions, where \(n_x\) and \(n_z\) denote proper integers. The time-evolutionary equations of \(z\) and \(\lambda\) are discretized into a forward difference equation, Eq. (15a) and a backward difference equation, Eq. (15c), respectively. Note that the boundary condition of Eq. (2) is also discretized and applied in Eq. (15a). Moreover, the equations obtained by discretizing Eqs. (14c) and (14e) are unified into Eq. (15c). The remaining stationary conditions of Eqs. (14d) and (14e) are also discretized and are represented in general forms in Eqs. (15d) and (15e), respectively.

For each \(k\) unknown parameters \(\hat{u}_k\) and \(\hat{\mu}_k\) for \(k = 1, \ldots, n_t\) be combined into the vector defined by

\[
\hat{U}(t) := [\hat{u}_1, \ldots, \hat{u}_{n_t}, \hat{\mu}_1, \ldots, \hat{\mu}_{n_t}]^{T}.
\]

Given the present state \(\hat{z}_1(t)\) and an initial solution \(\hat{U}(t)\), \(\hat{z}_k(t)\) is calculated iteratively from \(k = 1\) to \(k = n_t\) using Eq. (15a). Then, the terminal costate \(\hat{\lambda}_n(t)\) is determined by Eq. (15b). Next, we use Eq. (15c) to iteratively calculate \(\hat{\lambda}_k(t)\) for \(n_t = k\) to \(k = 1\). Since \(\hat{z}_k(t)\) and \(\hat{\lambda}_k(t)\) are determined by \(\hat{z}_1(t)\) and \(\hat{U}(t)\) through Eqs. (15a)–(15c),
Eqs. (15d) and (15e) can be regarded as a single equation
\[ F(\hat{U}(t), \hat{z}_1(t), t) := \left[ \hat{C}_1, \ldots, \hat{C}_n, \hat{E}_1, \ldots, \hat{E}_n \right] = 0 \in \mathbb{R}^n \]
where \( n_f := (n_x + c_x)n_t \). For a given \( \hat{z}_1(t) \) and \( \hat{U}(t), \hat{z}_1(t) \) and \( \hat{A}_1(t) \) are uniquely determined by Eqs. (15a)–(15c). Hence, \( \hat{z}_1(t) \) and \( \hat{A}_1(t) \) depend on \( \hat{z}_1(t) \) and \( \hat{U}(t) \), and consequently, it is reasonable to consider \( \hat{U}(t), \hat{z}_1(t), t \) as the argument of \( F \).

\( F \) is not necessarily equal to zero for any given \( \hat{z}_1(t) \) and \( \hat{U}(t) \). We use the norm \( \| F \| \) to evaluate the performance of optimality. The optimal control input is obtained by finding values for \( \hat{U}(t) \) that satisfy \( \| F \| = 0 \). Several algorithms have been developed that decrease the value of \( \| F \| \) by suitably updating \( \hat{U}(t) \).

### 4.1. C/GMRES method

A conventional method of updating \( \hat{U}(t) \) is to replace \( \hat{U}(t) \) with \( \hat{U}(t) + \alpha s \), where \( s \) is the steepest descent direction, and \( \alpha \) is the step length that satisfies the Armijo condition.\(^{12}\) The steepest descent direction approximates the direction as the gradient, while the Newton’s method uses the Hessian. However, these methods are computationally expensive, and it was shown that the C/GMRES algorithm\(^8\) is not only faster but also more numerically robust than such conventional algorithms. In the following, a brief description of the C/GMRES method applied to this problem is provided.

Instead of solving Eq. (16) itself at each time using an iterative method such as the steepest descent method or Newton’s method, we find the derivative of \( \hat{U}(t) \) with respect to time such that Eq. (16) is satisfied identically. Namely, we determine \( \hat{U}(t) \) such that
\[ F(\hat{U}(t), \hat{z}_1(t), t) = -\xi F(\hat{U}(t), \hat{z}_1(t), t) \]
is satisfied, where \( \xi \) is a positive constant introduced to stabilize \( F = 0 \). By total differentiation, we have
\[ \frac{\partial F}{\partial \hat{U}} \hat{U} = -\xi F - \frac{\partial F}{\partial \hat{z}_1} \hat{z}_1 - \frac{\partial F}{\partial t} \]
which can be regarded as a linear algebraic equation with coefficient matrix \( \frac{\partial F}{\partial \hat{U}} \), and can be used to determine \( \hat{U}(t) \) for given \( \hat{U}(t), \hat{z}_1(t), t \). Then, if the Jacobian \( \frac{\partial F}{\partial \hat{U}} \) is non-singular, we obtain for \( \hat{U}(t) \) the following differential equation.
\[ \hat{U} = \left( \frac{\partial F}{\partial \hat{U}} \right)^{-1} \left[ -\xi F - \frac{\partial F}{\partial \hat{z}_1} \hat{z}_1 - \frac{\partial F}{\partial t} \right]. \]

Instead of the iterative methods, the solution \( \hat{U}(t) \) of Eq. (16) can be updated by integrating Eq. (19) by a time marching method such as \( \hat{U}(t + \Delta t) \approx \hat{U}(t) + \hat{U}(t) \Delta t \). Since the solution curve \( \hat{U}(t) \) is traced by integrating a differential equation, this approach can be considered as a type of continuation method. From a computational viewpoint, the differential equation Eq. (19) involves expensive operations; i.e., the Jacobians \( \frac{\partial F}{\partial \hat{U}}, \frac{\partial F}{\partial \hat{z}_1} \) and \( \frac{\partial F}{\partial t} \) and the linear algebraic equation associated with \( \frac{\partial F}{\partial \hat{U}} \). To reduce computational cost, we employ two techniques: (i) the forward difference approximation is used for products of Jacobians and vectors to obtain a linear equation for \( \hat{U}(t) \), and (ii) the C/GMRES method is applied to solve the linear algebraic equation and update the solution. More detailed information about the implementation of C/GMRES is provided in Ref. 8).

### 4.2. Contraction mapping method

Recently, we have developed a more efficient algorithm than C/GMRES, called the contraction mapping method.\(^{10}\) Notably, our algorithm solves \( F(\hat{U}(t), \hat{z}_1(t), t) = 0 \) under the assumption that \( F(\hat{U}(t), \hat{z}_1(t), t) \) satisfies a particular structural condition with respect to \( \hat{U}(t) \). Compared with C/GMRES,\(^8\) the contraction mapping method\(^{10}\) has limited applicability, but requires less computational burden. In the following, we provide a brief description of the contraction mapping method.

Assume that \( F \) is given by
\[ F(\hat{U}(t), \hat{z}_1(t), t) = Q \hat{U}(t) - R(\hat{U}(t), \hat{z}_1(t), t) \]
where \( Q \in \mathbb{R}^{n \times m} \) is a non-singular constant matrix, and \( R \in \mathbb{R}^m \) is a vector-valued function. In the following, we consider the problem of solving Eq. (20) instead of Eq. (16).

Let \( P \in \mathbb{R}^m \) be defined by
\[ P(\hat{U}, \hat{z}_1, t) = Q^{-1} R(\hat{U}(t), \hat{z}_1(t), t). \]
Here, we adopt the following notations.
\[ P \circ P(\hat{U}, \hat{z}_1, t) = P(P(\hat{U}(t), \hat{z}_1(t), t), \hat{z}_1(t), t), \]
\[ P^k(\hat{U}, \hat{z}_1, t) = P \circ \cdots \circ P(\hat{U}, \hat{z}_1, t). \]
Suppose that \( \hat{U}(t) \) is updated as
\[ \hat{U}(t) = P^k(\hat{U}(t - \Delta t), \hat{z}_1(t), t) \]
where \( t = \Delta t, 2\Delta t, \ldots, j\Delta t \) for \( j \in \mathbb{N}_+ \) and \( k \in \mathbb{N}_+ \) is a design parameter. In Ref. 10, we showed under some assumptions that \( \| F \| \) is ultimately bounded and is monotonically decreasing whenever \( \| F \| > \varepsilon \). We also showed that the upper bound \( \varepsilon \) of \( \| F \| \) converges to zero as \( k \) increases to infinity. Hence, in the contraction mapping method, by selecting design parameter \( k \), we can achieve a satisfactory trade-off between computational burden and error performance. More detailed information about the contraction mapping method is provided in Ref. 10).

### 5. Illustrative Example

In this section, we provide an illustrative example to verify the effectiveness of the proposed method. Let \( n = 2 \) and \( h = 1 \). Thus, we consider the fluid flow described by the Burgers’ equation for the two-dimensional square domain \( x \in \Omega := [0 \ 1] \times [0 \ 1] \). In this example, we consider the control problem of fluid dynamics to achieve the desired flow using boundary control inputs. For this purpose, let \( \phi \) and \( L \) in the performance index of Eq. (5) be set as
\[ \phi = \frac{1}{2} \int_\Omega \left[ z(x, t + T) - z_f \right] W_1 \left[ z(x, t + T) - z_f \right] dx. \]
$$L = \frac{1}{2} \int_{\Omega} \left[ z(x, \tau) - z_f \right] W_2 \left[ z(x, \tau) - z_f \right] dx + \sum_{i=1}^{2} \int_{\partial \Omega} u_i^r W_3 \left[ z(x, \tau) - z_f \right] d\sigma$$

where $z_f$ denotes the desired final state. Here, we set the initial state $z_0$ and the desired state $z_f$ as

$$z_0 = \left[ -\cos(\pi x_1) \sin(\pi x_2) \right] \quad \text{and} \quad z_f = \left[ 0.1 \quad 0.1 \right].$$

In this case, we obtain the following stationary conditions that must be satisfied for the above performance index to be minimized. For $x \in \Omega$, we have

$$\frac{\partial z}{\partial t} = v \nabla^2 z - (z \cdot \nabla)z, \quad \frac{\partial \lambda}{\partial t} = -W_2(z - z_f) - v \nabla^2 \lambda - (z \cdot \nabla)\lambda$$

$$+ \begin{bmatrix} A'z_{1i} \\ A'z_{2i} \end{bmatrix} - \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \begin{bmatrix} z_{1i} \\ z_{2i} \end{bmatrix}$$

and for $i = 1, 2$ and $x_i \in \partial \Omega$, we have

$$\begin{bmatrix} z_{1i} A_1 + v A_{1i} \\ z_{2i} A_2 + v A_{2i} \end{bmatrix} = 0 \quad \text{(23d)}$$

$$\begin{bmatrix} W_3 u_i + v s(x_i) A' G_i \end{bmatrix} = 0. \quad \text{(23e)}$$

Here, the Crank-Nicolson finite-difference approximation method can be applied to discretize the above stationary conditions. Furthermore, note that the optimality condition of Eq. (23e) can be reduced into the same form as Eq. (20). Hence, we can solve the optimization problem by applying the contraction mapping method. We choose $G(x)$ so that the control inputs are employed at the points $(x_1, x_2) = (0, 0), (1, 0), (0, 1), (1, 1)$. Owing to the initialization of the optimal solution $\hat{U}(0)$, the length of the horizon is selected such that $T(0) = 0.05$ as $t \to \infty$; that is, $T(t) = 0.05(1 - e^{-0.5t})$. Other parameters employed in the numerical simulations are as follows: $W_1 = \text{diag}(10, 10), \ W_2 = \text{diag}(10^5, 10^5), \ W_{31} = W_{32} = \text{diag}(1, 1), \ \Delta t = 0.01, \ n_x = 100, \ n_t = 10, \ k = 1$ and $v = 0.05$. In Figs. 3–7, the dashed and solid arrows show the time history of the state $z(x, t)$ of the system without

![Fig. 3. Time history of the state $z(x, t)$ at $t = 0.$](image1)

![Fig. 4. Time history of the state $z(x, t)$ at $t = 2.$](image2)

![Fig. 5. Time history of the state $z(x, t)$ at $t = 4.$](image3)

![Fig. 6. Time history of the state $z(x, t)$ at $t = 6.$](image4)
control inputs and with control inputs, respectively. The figures reveal the effectiveness of the proposed method. Figures 8 and 9 show the time history of the control inputs. Figures 10 and 11 show the time histories of the state error and optimality error, respectively. It was verified that the state error and the optimality error successfully converge to zero. A simulation is performed on a laptop computer (Panasonic CF-S9; CPU: Intel(R) Core(TM) i5; 2.4 [GHz]; memory: 3.4 [GB]; OS: Windows 7, 32 bit; software: Matlab). The average computational time per update (1 control cycle) is 36.2 [ms].

6. Conclusions
In this study, first we formulated a framework for designing receding horizon controllers for high-dimensional Burgers’ equations subject to constraints. Using the variational principle, we derived the stationary conditions that must be satisfied to optimize the control performance over a finite horizon. Next, we reduced these stationary conditions to finite difference equations that can be solved by a numerical algorithm. Then, we showed that the C/GMRES and contraction mapping methods can be applied to solve the stationary conditions obtained for the optimization problem of the high-dimensional Burgers’ equation. Finally, the effectiveness of the proposed method was verified by numerical simulation.
Acknowledgments

This work was supported in part by Japan Society for the Promotion of Science, Grant-in-Aid for Young Scientists (24760334).

References


