Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Efficient Solution Method Based on Inverse Dynamics of Optimal Control Problems for Fixed-Based Rigid-Body Systems [⋆]

S. Katayama [*] T. Ohtsuka [*]

[*] *Department of Systems Science, Graduate School of Informatics, Kyoto University, Sakyo-ku, Kyoto, 606-8501, Japan*

Abstract: We propose an efficient solution method of finite horizon optimal control problems (FHOCPs) for fixed-based rigid-body systems based on inverse dynamics. Our method can reduce the computational cost compared with the conventional FHOCP based on forward dynamics. We reformulate the FHOCP for the rigid-body systems by utilizing the generalized acceleration as the decision variables and inverse dynamics as the equality constraint. We derive the necessary conditions of the optimal control, namely, the optimality conditions, and formulate a two-point boundary-value problem that can be solved efficiently by using the recursive Newton Euler algorithm (RNEA) and the partial derivatives of RNEA. The results of the several numerical experiments on nonlinear model predictive control using the proposed formulation demonstrate the effectiveness of our approach.

Keywords: Nonlinear Control, Optimal Control, Robotics

1. INTRODUCTION

Nonlinear model predictive control (NMPC) (Magni et al. (2008)) has attracted much attention in various fields, especially in the robotics community in which it has been applied to wheeled robots (Hsieh and Liu (2012)), multicopters (Kamel et al. (2015)), and legged robots (Koenemann et al. (2015); Neunert et al. (2018)). This is because NMPC achieves a state-feedback control law treating nonlinear dynamics and constraints explicitly by solving a finite horizon optimal control problem (FHOCP) at each sampling time. However, we cannot implement NMPC unless we can solve the FHOCP within a given sampling period. From this view, we still need to reduce the computational cost of NMPC when we apply it to complicated rigid-body systems such as legged robots that have a high dimensional state, complicated dynamics, and require a short sampling period.

When solving an FHOCP for NMPC, i.e., finding the optimal control input that minimizes the performance index subject to the dynamics of a controlled system, the most efficient algorithms of NMPC belong to the first-order methods (Graichen and Käpernick (2012)) or the second-order methods (Ohtsuka (2004); Todorov and Li (2005); Diehl et al. (2005); Ferreau et al. (2014); Frasch et al. (2015); Zanelli et al. (2020)), both of which need to compute a function representing the system's dynamics and its partial derivatives. In the FHOCP for a rigid-body system, the state is composed of the generalized configuration and velocity, and the behavior of the system's dynamics for given decision variables is simulated over the horizon by integrating the generalized acceleration. Previous studies applying NMPC to such systems, e.g., Diehl et al. (2006), Gerdts et al. (2012), Koenemann et al. (2015), and Neunert et al. (2018), assume the generalized torque as decision variables of the FHOCP and compute the generalized acceleration from the given generalized configuration, velocity, and torque, which is called forward dynamics. These studies also compute the partial derivatives of the function of forward dynamics with respect to the generalized configuration, velocity, and torque to apply the aforementioned NMPC algorithms. Forward dynamics and the partial derivatives are so complicated that their computations occupy most of the total computational cost of NMPC, requiring efficient numerical methods to compute them. Featherstone (1983, 2008) proposed the articulated body algorithm (ABA), a highly efficient recursive algorithm to compute the forward dynamics of open-chain rigid-body systems whose degree of freedom is as large as realistic robots such as manipulators and humanoid robots. When it comes to the partial derivatives of the function of forward dynamics, a recursive algorithm proposed by Carpentier and Mansard (2018) is faster than other methods such as finite-difference approximation or automatic differentiation (Neunert et al. (2016)). However, it still takes up a lot of computational time, which makes it difficult to use with NMPC.

An alternative representation of the dynamics of rigid-body systems is inverse dynamics, which involves the calculation of the generalized torque for the given generalized configuration, velocity, and acceleration. Like forward dynamics, inverse dynamics and the partial derivatives of the function of inverse dynamics are complicated and require efficient numerical methods. The recursive Newton Euler algorithm (RNEA) (Featherstone (2008)) is the most efficient algorithm to compute inverse dynamics. To calculate the partial derivatives of the function of inverse dynamics, a recursive algorithm proposed by Carpentier

[⋆] This work was partly supported by JSPS KAKENHI Grant Number 15H02257.

and Mansard (2018), referred to as the partial derivatives of RNEA in their paper, is also more efficient than other methods such as finite-difference and automatic differentiation (Neunert et al. (2016)).

Featherstone (2008) demonstrated through analysis of arithmetic operations and numerical experiments that the computational cost of RNEA is less than that of ABA. Furthermore, the computational cost of the partial derivatives of RNEA is less than that of the recursive algorithm for the partial derivatives of the function of forward dynamics proposed by Carpentier and Mansard (2018). The automatic differentiation of the function of inverse dynamics is also faster than that of forward dynamics (Neunert et al. (2016)). Therefore, we expect to reduce the computational cost by replacing forward dynamics and the partial derivatives of the function of forward dynamics in the FHOCP with inverse dynamics and the partial derivatives of the function of inverse dynamics. Fortunately, this is possible because the forward and inverse dynamics of a fully actuated system represent the equivalent constraints derived from the same equation of motion.

Previous works on the FHOCP based on inverse dynamics instead of forward dynamics are found in the context of direct trajectory optimization with contacts (Erez and Todorov (2012); Posa et al. (2014)). However, these studies focus on the stable solution method for the complementarity problem arising from contacts with the environment rather than the computational efficiency. As a result, they use the direct multiple shooting, where all variables are treated as the decision variables of the optimization problem. Furthermore, they do not utilize efficient algorithms for rigid body dynamics such as RNEA.

In this study, we propose an efficient solution method of the FHOCP for rigid body systems using RNEA and the partial derivatives of RNEA, which reduces the computational cost compared with the conventional FHOCP based on forward dynamics. We reformulate the FHOCP with the generalized acceleration utilized as decision variables and inverse dynamics as an equality constraint. We then derive the necessary conditions of the optimal control, namely, the optimality conditions, and formulate a two-point boundary-value problem (TPBVP) that can be efficiently solved by RNEA and the partial derivatives of RNEA. For the resultant TPBVP, we can use either the single shooting method or the multiple shooting method with condensing (Bock and Plitt (1984)), whose optimization problems are smaller than those of direct multiple shooting. The TPBVP can also be combined with NMPC algorithms such as Ohtsuka (2004); Diehl et al. (2005); Ferreau et al. (2014); Frasch et al. (2015); Zanelli et al. (2020). We conduct numerical experiments in several problem settings and show that the FHOCP based on inverse dynamics is faster than the FHOCP based on forward dynamics.

This paper is composed as follows. In Section 2, we introduce the dynamics of rigid-body systems, and efficient algorithms to compute them. In Section 3, we reformulate the FHOCP and derive the optimality conditions based on inverse dynamics and apply it to NMPC. In Section 4, we show a numerical simulation of NMPC and show that the proposed method reduces the computational cost compared with the conventional formulation based on forward dynamics. In Section 5, we conclude our paper.

2. RIGID-BODY DYNAMICS

2.1 Dynamics of the Rigid-Body Systems

In this paper, we consider a fully actuated open-chain fixed-based rigid-body system that has n joints. Let $q \in \mathbb{R}^n$ be the generalized configuration of the system and $\tau \in \mathbb{R}^n$ be the generalized torque of the system. The equation of the motion of this system is given by

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) = \tau, \tag{1}$$

where $M(q) \in \mathbb{R}^{n \times n}$ denotes the inertial matrix, $C(q, \dot{q})\dot{q} \in \mathbb{R}^n$ encompasses the Coriolis, frictional, and centrifugal forces, and $g(q) \in \mathbb{R}^n$ denotes the gravity torque. Forward dynamics is a calculation of the generalized acceleration $\ddot{q}$ under the given generalized configuration q, generalized velocity $\dot{q}$, and generalized torque τ. In the following, we describe the function of forward dynamics as $\mathrm{FD}(\cdot, \cdot, \cdot) : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$, e.g.,

$$\mathrm{FD}(q, \dot{q}, \tau) = M^{-1}(q)(\tau - C(q, \dot{q})\dot{q} - g(q)). \tag{2}$$

On the other hand, inverse dynamics is a calculation of the generalized torque τ under the given generalized configuration q, velocity $\dot{q}$, and acceleration $\ddot{q}$. In the following, we describe the function of inverse dynamics as $\mathrm{ID}(\cdot, \cdot, \cdot) : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$, e.g.,

$$\mathrm{ID}(q, \dot{q}, \ddot{q}) = M(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q). \tag{3}$$

The constraints

$$\ddot{q} - \mathrm{FD}(q, \dot{q}, \tau) = 0, \tag{4}$$

and

$$\tau - \mathrm{ID}(q, \dot{q}, \ddot{q}) = 0, \tag{5}$$

are equivalent because the system is fully actuated, i.e., the dimensions of $\ddot{q}$ and that of τ are the same and $M(q)$ is square and non-singular.

2.2 Algorithms of the Rigid-Body Dynamics

When the system has three or more joints, it is difficult to write the equation of motion (1) explicitly because it is too complicated. In such cases, recursive algorithms are required to compute forward and inverse dynamics. ABA (Featherstone (1983, 2008)) would be the most efficient algorithm to compute forward dynamics, and RNEA (Featherstone (2008)) is the fastest algorithm to compute inverse dynamics. More detailed descriptions of rigid-body algorithms are found in Featherstone (2008). In this paper, we assume that we compute $\mathrm{FD}(q, \dot{q}, \tau)$ using ABA and $\mathrm{ID}(q, \dot{q}, \ddot{q})$ using RNEA.

2.3 Algorithms of the Partial Derivatives of the Rigid-Body Dynamics

The most efficient algorithms of NMPC and nonlinear optimal control are classified into first-order methods and second-order methods, in which we need to compute the partial derivatives of the state equations and constraints with respect to the state and control input. If the controlled systems are rigid-body systems, we need to compute the partial derivatives of $\mathrm{FD}(q, \dot{q}, \tau)$ or $\mathrm{ID}(q, \dot{q}, \ddot{q})$.

Explicit formulations of their derivatives are also difficult because of their extreme complexities. Carpentier and Mansard (2018) proposes an efficient algorithm of the partial derivatives of $\mathrm{ID}(q, \dot{q}, \ddot{q})$, called the partial derivatives of the RNEA. They also proposes an efficient way to compute the partial derivatives of $\mathrm{FD}(q, \dot{q}, \tau)$. Because they are faster than other methods such as finite-difference or automatic differentiation (Neunert et al. (2016)), we utilize their algorithms to compute the partial derivatives of $\mathrm{FD}(q, \dot{q}, \tau)$ and $\mathrm{ID}(q, \dot{q}, \ddot{q})$ in this paper.

3. OPTIMAL CONTROL PROBLEM BASED ON INVERSE DYNAMICS

The forward dynamics-based FHOCP for the rigid body systems is a problem to find the optimal control input minimizing a performance index subject to the state equation

$$\frac{d}{dt}\begin{bmatrix} q \\ \dot{q} \end{bmatrix} := \begin{bmatrix} \dot{q} \\ \mathrm{FD}(q, \dot{q}, \tau) \end{bmatrix}. \tag{6}$$

In solving the FHOCP, $\mathrm{FD}(\cdot,\cdot,\cdot)$, $\frac{\partial \mathrm{FD}}{\partial q}(\cdot,\cdot,\cdot)$, $\frac{\partial \mathrm{FD}}{\partial \dot{q}}(\cdot,\cdot,\cdot)$, and $\frac{\partial \mathrm{FD}}{\partial \tau}(\cdot,\cdot,\cdot)$ are computed as many times as the division number of the finite horizon to predict the future trajectory of the system and evaluate its sensitivity. In this section, we reformulate the FHOCP based on inverse dynamics and its partial derivatives. We then derive the necessary conditions of the optimal control and formulate the TPBVP based on inverse dynamics.

3.1 Formulation of the FHOCP and the derivation of the optimality conditions

To derive the optimality conditions based on inverse dynamics, we consider the generalized acceleration as the decision variables in the FHOCP. Note that the generalized acceleration $\ddot{q}$ and torque τ of a fully actuated system have the same dimensions and are equivalent through $\ddot{q} = \mathrm{FD}(q, \dot{q}, \tau)$ and $\tau = \mathrm{ID}(q, \dot{q}, \ddot{q})$ as well as the constraints (4) and (5). We set the state as $x := [q^{\mathrm{T}} \ \dot{q}^{\mathrm{T}}]^{\mathrm{T}}$. Since we assume that the generalized acceleration $\ddot{q}$ $(t_0 \leq t' \leq t_f)$ is known, the state equation is simply given by

$$\frac{d}{dt}\begin{bmatrix} q \\ \dot{q} \end{bmatrix} := f(\dot{q}, \ddot{q}) = \begin{bmatrix} \dot{q} \\ \ddot{q} \end{bmatrix}. \tag{7}$$

However, (7) cannot consider the effect of the system's dynamics (1). We therefore consider the inverse dynamics (3) as an equality constraint in the FHOCP. We also assume that q_0 and v_0 are given as

$$q_0 = q(t_0), \ v_0 = \dot{q}(t_0). \tag{8}$$

The optimization problem is then given as follows: find the optimal generalized acceleration $\ddot{q}(t')$ $(t_0 \leq t' \leq t_f)$ minimizing the cost function

$$J = \varphi(q(t_f), \dot{q}(t_f)) + \int_{t_0}^{t_f} L(q(t'), \dot{q}(t'), \ddot{q}(t'), u(t'))dt', \tag{9}$$

subject to (7) and

$$u(t') - \mathrm{ID}(q(t'), \dot{q}(t'), \ddot{q}(t')) = 0. \tag{10}$$

Note that u denotes the control input torques and we define the stage cost as $L(q, \dot{q}, \ddot{q}, u)$ so that the acceleration can be included in its arguments to evaluate the acceleration in the cost function explicitly. In contrast, the stage cost in the forward dynamics-based formulation is often formulated by a function of the configuration, velocity, and torques, i.e., $L(q, \dot{q}, u)$. This FHOCP is then equivalent to the FHOCP based on forward dynamics unless the terms related to the generalized acceleration are included in the cost function (9). However, we sometimes need to impose the penalty on the generalized acceleration in the cost function to improve the numerical stability in practice.

Next, we discretize the FHOCP for numerical computation. We devide the horizon into N steps and introduce $\Delta\tau := (t_f - t_0)/N$. We discretize the generalized configuration into q_0, ..., q_N, velocity into v_0, ..., v_N, acceleration into a_0, ..., a_{N-1}, and the control input into u_0, ..., u_{N-1}. The cost function (9) is then discretized as

$$J = \varphi(q_N, v_N) + \sum_{i=0}^{N-1} L(q_i, v_i, a_i, u_i)\Delta\tau, \tag{11}$$

the state equation (7) is discretized as

$$\begin{bmatrix} q_{i+1} \\ v_{i+1} \end{bmatrix} = \begin{bmatrix} q_i \\ v_i \end{bmatrix} + \begin{bmatrix} v_i \\ a_i \end{bmatrix}\Delta\tau, \ \ i = 0, ..., N-1, \tag{12}$$

and the constraint (10) is discretized as

$$u_i - \mathrm{ID}(q_i, v_i, a_i) = 0, \ \ i = 0, ..., N-1. \tag{13}$$

To derive the optimality conditions, we introduce the Lagrange multipliers for (12), $\lambda_1, ..., \lambda_N \in \mathbb{R}^n, \gamma_1, ..., \gamma_N \in \mathbb{R}^n$, and the Lagrange multipliers for (13), $\beta_0, ..., \beta_{N-1} \in \mathbb{R}^n$ and formulate the augmented cost function

$$\begin{aligned} \tilde{J} = \varphi(q_N, v_N) &+ \sum_{i=0}^{N-1} L(q_i, v_i, a_i, u_i)\Delta\tau \\ &+ \sum_{i=0}^{N-1} \lambda_{i+1}^{\mathrm{T}} (v_i \Delta\tau + q_i - q_{i+1}) \\ &+ \sum_{i=0}^{N-1} \gamma_{i+1}^{\mathrm{T}} (a_i \Delta\tau + v_i - v_{i+1}) \\ &+ \sum_{i=0}^{N-1} \beta_i^{\mathrm{T}} (u_i - \mathrm{ID}(q_i, v_i, a_i)) \Delta\tau. \end{aligned} \tag{14}$$

By making all variables perturbed, we obtain $\delta\tilde{J}$ as in (15). We then derive the optimality conditions, necessary conditions for $\delta\tilde{J} = 0$ under arbitrary $\delta q_1, ..., \delta q_N$, $\delta v_1, ..., \delta v_N$, $\delta a_0, ..., \delta a_{N-1}$, and $\delta u_0, ..., \delta u_{N-1}$ (Note that $\delta q_0 = 0$ and $\delta v_0 = 0$ because q_0 and v_0 are fixed by (8)):

$$\left(\frac{\partial \varphi}{\partial q}\right)^{\mathrm{T}} (q_N, v_N) - \lambda_N = 0, \tag{16}$$

$$\left(\frac{\partial L}{\partial q}\right)^{\mathrm{T}} (q_i, v_i, a_i, u_i)\Delta\tau - \left(\frac{\partial \mathrm{ID}}{\partial q}\right)^{\mathrm{T}} (q_i, v_i, a_i) \beta_i \Delta\tau + \lambda_{i+1} - \lambda_i = 0, \tag{17}$$

for $i = 1, ..., N-1$,

$$\left(\frac{\partial \varphi}{\partial \dot{q}}\right)^{\mathrm{T}} (q_N, v_N) - \gamma_N = 0, \tag{18}$$

$$\left(\frac{\partial L}{\partial \dot{q}}\right)^{\mathrm{T}} (q_i, v_i, a_i, u_i)\Delta\tau - \left(\frac{\partial \mathrm{ID}}{\partial \dot{q}}\right)^{\mathrm{T}} (q_i, v_i, a_i) \beta_i \Delta\tau + \lambda_{i+1}\Delta\tau + \gamma_{i+1} - \gamma{=}0, \tag{19}$$

for $i = 1, ..., N-1$,

$$
\begin{aligned}
\delta \tilde{J} = &\left(\frac{\partial \varphi}{\partial q}\right)^{\mathrm{T}} (q_N, v_N) \delta q_N + \left(\frac{\partial \varphi}{\partial \dot{q}}\right)^{\mathrm{T}} (q_N, v_N) \delta v_N \\
&+ \sum_{i=0}^{N-1} \left\{ \left(\frac{\partial L}{\partial q}\right)^{\mathrm{T}} (q_i, v_i, u_i) \delta q_i + \left(\frac{\partial L}{\partial \dot{q}}\right)^{\mathrm{T}} (q_i, v_i, u_i) \delta v_i + \left(\frac{\partial L}{\partial \ddot{q}}\right)^{\mathrm{T}} (q_i, v_i, u_i) \delta a_i + \left(\frac{\partial L}{\partial u}\right)^{\mathrm{T}} (q_i, v_i, u_i) \delta u_i \right\} \Delta \tau \\
&+ \sum_{i=0}^{N-1} \lambda_{i+1}^{\mathrm{T}} \left(\delta v_i \Delta \tau + \delta q_i - \delta q_{i+1}\right) + \sum_{i=0}^{N-1} \gamma_{i+1}^{\mathrm{T}} \left(\delta a_i \Delta \tau + \delta v_i - \delta v_{i+1}\right) \\
&+ \sum_{i=0}^{N-1} \beta_i^{\mathrm{T}} \left\{ \delta u_i - \left(\frac{\partial \mathrm{ID}}{\partial q}\right) (q_i, v_i, a_i)\, \delta q_i - \left(\frac{\partial \mathrm{ID}}{\partial \dot{q}}\right) (q_i, v_i, a_i)\, \delta v_i - \left(\frac{\partial \mathrm{ID}}{\partial \ddot{q}}\right) (q_i, v_i, a_i)\, \delta a_i \right\} \Delta \tau
\end{aligned} \tag{15}
$$

$$
\gamma_{i+1} + \left(\frac{\partial L}{\partial \ddot{q}}\right)^{\mathrm{T}} (q_i, v_i, a_i, u_i) - \left(\frac{\partial \mathrm{ID}}{\partial \ddot{q}}\right)^{\mathrm{T}} (q_i, v_i, a_i) \beta_i = 0, \tag{20}
$$

for $i = 0, ..., N-1$, and

$$
\left(\frac{\partial L}{\partial u}\right)^{\mathrm{T}} (q_i, v_i, a_i, u_i) + \beta_i = 0, \tag{21}
$$

for $i = 0, ..., N-1$. From (21), we can eliminate $\beta_1, ..., \beta_N$ in (17), (19) and (20) as

$$
\begin{aligned}
&\left(\frac{\partial \mathrm{ID}}{\partial q}\right)^{\mathrm{T}} (q_i, v_i, a_i) \left(\frac{\partial L}{\partial u}\right)^{\mathrm{T}} (q_i, v_i, a_i, u_i) \Delta \tau \\
&\quad + \left(\frac{\partial L}{\partial q}\right)^{\mathrm{T}} (q_i, v_i, a_i, u_i) \Delta \tau + \lambda_{i+1} - \lambda_i = 0,
\end{aligned} \tag{22}
$$

$$
\begin{aligned}
&\left(\frac{\partial \mathrm{ID}}{\partial \dot{q}}\right)^{\mathrm{T}} (q_i, v_i, a_i) \left(\frac{\partial L}{\partial u}\right)^{\mathrm{T}} (q_i, v_i, a_i, u_i) \Delta \tau \\
&\quad + \left(\frac{\partial L}{\partial \dot{q}}\right)^{\mathrm{T}} (q_i, v_i, a_i, u_i) \Delta \tau + \lambda_{i+1} \Delta \tau + \gamma_{i+1} - \gamma_i = 0,
\end{aligned} \tag{23}
$$

and

$$
\begin{aligned}
&\gamma_{i+1} + \left(\frac{\partial L}{\partial \ddot{q}}\right)^{\mathrm{T}} (q_i, v_i, a_i, u_i) \\
&\quad + \left(\frac{\partial \mathrm{ID}}{\partial \ddot{q}}\right)^{\mathrm{T}} (q_i, v_i, a_i) \left(\frac{\partial L}{\partial u}\right)^{\mathrm{T}} (q_i, v_i, a_i, u_i) = 0,
\end{aligned} \tag{24}
$$

respectively. The FHOCP is then reduced to the following nonlinear problem: find the sequence of the optimal generalized acceleration $a_0, ..., a_{N-1}$ satisfying (8), (12), (16), (18), and (22)–(24). Note that under the given $a_0, ..., a_{N-1}$, we can first determine $v_0, ..., v_N$ and $q_0, ..., q_N$ from (8) and (12). Then, we can determine $u_0, ..., u_{N-1}$ by

$$
u_i = \mathrm{ID}(q_i, v_i, a_i). \tag{25}
$$

Finally, we can determine the multipliers $\lambda_N, ..., \lambda_1$ from (16) and (22) and $\gamma_N, ..., \gamma_1$ from (18) and (23). The errors from the optimal control of given $a_0, ... a_{N-1}$ can be then measured by (24) for $i = 0, ..., N-1$, which defines the TPBVP.

Except the terms related to the acceleration in (11), the proposed formulation is identical to the formulation based on forward dynamics because the adjointed constraints (12) and (13) are equivalent to the discretized state equation (6), which is treated as a constraints in the FHOCP based on forward dynamics,

$$
\begin{bmatrix} q_{i+1} \\ v_{i+1} \end{bmatrix} = \begin{bmatrix} q_i \\ v_i \end{bmatrix} + \begin{bmatrix} v_i \\ \mathrm{FD}(q_i, v_i, u_i) \end{bmatrix} \Delta \tau, \quad i = 0, ..., N-1. \tag{26}
$$

The dimensions of the decision variables are the same between the two formulations, i.e., the total dimension of $a_0, ..., a_{N-1}$ and that of $u_0, ..., u_{N-1}$ are the same.

3.2 Application to NMPC

Next, we apply the proposed formulation of the FHOCP to NMPC for rigid-body systems. We hereafter consider the optimality-conditions-based numerical methods of NMPC, i.e., numerical methods that seek a solution satisfying a number of conditions such as the optimality conditions, the Pontryagin's maximum principle (PMP), or the Karush-Kuhn-Tucker (KKT) conditions. We also assume that these methods are Hessian-free, that is, we do not need to compute further derivatives of these conditions explicitly. It is worth noting that typical efficient second-order algorithms of NMPC use Hessian approximation such as finite-difference (Ohtsuka (2004)) or Gauss-Newton (Diehl et al. (2005); Ferreau et al. (2014); Frasch et al. (2015); Zanelli et al. (2020)) because computing further derivatives requires too much computational time and is impractical.

In NMPC for rigid-body systems based on the forward dynamics-based FHOCP, we compute the optimal control input torques from the current time t to the finite future $t + T$ $(T > 0)$ on the basis of the current state $q(t)$ and $\dot{q}(t)$ at each sampling time. Then, the initial value of the optimal control input torques is applied to the actual system. In contrast, the solution of the proposed FHOCP based on inverse dynamics is the optimal generalized acceleration $a_0, ..., a_{N-1}$. Therefore, after obtaining the optimal generalized acceleration $a_0, ..., a_{N-1}$ by solving the proposed FHOCP on the basis of the current state, we have to compute the actual control input applied to the system, e.g., by

$$
u(t_{\mathrm{app}}) = \mathrm{ID}(\tilde{q}(t_{\mathrm{app}}), \tilde{\dot{q}}(t_{\mathrm{app}}), a_0), \tag{27}
$$

where t_{app} is the instant when the optimal control input is applied to the system and $\tilde{q}(t_{\mathrm{app}})$ and $\tilde{\dot{q}}(t_{\mathrm{app}})$ are the estimations of the generalized configuration and generalized velocity at t_{app}. Note that if a lag between the state measurement and the application of the control input is sufficiently short, i.e., $t \simeq t_{\mathrm{app}}$, we can estimate $\tilde{q}(t_{\mathrm{app}}) \simeq q(t)$ and $\tilde{\dot{q}}(t_{\mathrm{app}}) \simeq \dot{q}(t)$. When we cannot disregard the lag, we estimate $\tilde{q}(t)$ and $\tilde{\dot{q}}(t)$ using the optimal generalized acceleration computed at the previous sampling time, $\hat{a}_0, ..., \hat{a}_{N-1}$, e.g., by

$$
\tilde{q}(t_{\mathrm{app}}) = q(t) + (t_{\mathrm{app}} - t) \dot{q}(t), \; \tilde{\dot{q}}(t_{\mathrm{app}}) = \dot{q}(t) + (t_{\mathrm{app}} - t) \hat{a}_0. \tag{28}
$$

4. NUMERICAL EXPERIMENTS

In this section, we compare the proposed FHOCP based on inverse dynamics with the FHOCP based on forward dynamics in terms of computational time by simulating NMPC using these FHOCPs.

4.1 NMPC controllers using the continuation/GMRES method

In numerical experiments of NMPC in this paper, we utilize the continuation/GMRES (C/GMRES) method (Ohtsuka (2004)) as an efficient Hessian-free method of NMPC. The C/GMRES method achieves fast computation by tracking the solution of the FHOCP, i.e., it computes the time variant of the optimal solution instead of computing it directly by solving the FHOCP. Let $U(t)$ be a solution of the FHOCP and $F(U(t), x(t), t) = 0$ be the equation $U(t)$ has to satisfy, i.e., $U(t) = [u_0^{\mathrm{T}} \cdots u_{N-1}^{\mathrm{T}}]^{\mathrm{T}}$ for the FHOCP based on forward dynamics and $U(t) = [a_0^{\mathrm{T}} \cdots a_{N-1}^{\mathrm{T}}]^{\mathrm{T}}$ for the FHOCP based on inverse dynamics. $F(U(t), x(t), t)$ for the FHOCP based on forward dynamics is easily obtained by using the state equation (6) in the TPBVP described in Ohtsuka (2004). $F(U(t), x(t), t)$ for the FHOCP based on forward dynamics is composed by (24) for $i = 0, ..., N-1$. Note that the current state $x(t)$ is given by $[q(t)^{\mathrm{T}}\ v(t)^{\mathrm{T}}]^{\mathrm{T}}$. The C/GMRES method does not solve the nonlinear equation $F(U(t), x(t), t) = 0$ directly but solves the following equation that is derived using the continuation method (Richter and DeCarlo (1983))

$$\frac{\partial F}{\partial U}\dot{U} = -\frac{\partial F}{\partial x}\dot{x} - \frac{\partial F}{\partial t} - \zeta F, \tag{29}$$

where $\zeta > 0$ is a stabilization parameter. Note that we omit arguments in (29). The products of the partial derivatives of F and vectors in (29) are computed by the finite-difference approximation and the partial derivatives of F are not computed explicitly, i.e., we do not need to compute the partial derivatives of the state equation, the constraints, and the cost functions twice, which means it is a Hessian-free method. The C/GMRES method computes $\dot{U}$ by solving the linear problem (29) using the GMRES method (Kelly (1995)), a fast inexact numerical solver of the linear problem, and updates the solution by

$$U(t + \Delta t) = U(t) + \dot{U}\Delta t, \tag{30}$$

where $\Delta t > 0$ is the sampling period.

In the following, we consider two NMPC controllers for fully actuated rigid-body systems: one utilizing the C/GMRES method solving the FHOCP based on forward dynamics and the other utilizing the C/GMRES method solving the FHOCP based on inverse dynamics. We call the former C/GMRES (FD) and the latter C/GMRES (ID). Note that in C/GMRES (ID), the sequence of the optimal generalized acceleration is updated by (30). The control input to the system is then obtained by (27) and the lag in the state measurement is coped with by (28) with setting $t_{\mathrm{app}} = t + \Delta t$. The following describes the common settings of the two controllers. We set the length of the horizon of NMPC as a time-dependent smooth function $T(t)$ such that $T(0) = 0$ and $T(t) \to T_f$ $(t \to \infty)$, e.g., as

$$T(t) = T_f(1 - e^{-\alpha t}), \tag{31}$$

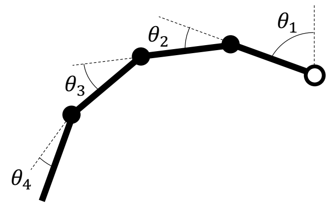

Fig. 1. A fixed-based fully actuated four-link arm

for the initialization of the solution (see Ohtsuka (2004)), where $T_f = 0.5, \alpha = 1.0$. We also set the increment of the finite-difference approximation in the C/GMRES method as $h = 1.0 \times 10^{-8}$ and the stabilization parameters of the C/GMRES method ζ by the reciprocal of the sampling period. The remaining parameters of the C/GMRES method are the number of the discretization of the horizon N and the number of the iteration of the GMRES method $k_{\max}$, which we do not fix here. Note that the computational time of the C/GMRES method is determined by the number of joints n, N, and $k_{\max}$ with n indicating the complexity of the system. As n increases, the computational time of RNEA, ABA, and the partial derivatives of both functions of inverse and forward dynamics increase. As N increases, the accuracy of the solution increases because the approximation of the continuous FHOCP by the discretized FHOCP becomes more accurate and the computational cost increases because the number of times to calculate RNEA, ABA, and both partial derivatives increases. As $k_{\max}$ increases, the accuracy of the solution and the computational cost also increases because the linear problem (29) is solved more accurately.

4.2 Swing-Up Control of a Fixed-Based Fully Actuated Four-link Arm

Problem Settings First, we simulate the swing-up control of a fully actuated four-link arm depicted in Fig. 1 using C/GMRES (FD) and C/GMRES (ID). In Fig. 1, a white circle is the fixed-base joint and the black circles are the other joints. The generalized configuration is given by $q = [\theta_1\ \theta_2\ \theta_3\ \theta_4]^{\mathrm{T}}$. We assume that each joint has no mass and inertia and that each link has the same physical characteristics. We set the whole length of the link to 1[m], the width to 0.1[m], and the mass to 1[kg]. We also assume that the mass is distributed uniformly in the link. We construct the terminal cost of both C/GMRES (FD) and C/GMRES (ID) by

$$\varphi(q, v) = \frac{1}{2}(q - q_{\mathrm{ref}})^{\mathrm{T}} Q_q (q - q_{\mathrm{ref}}) + \frac{1}{2}v^{\mathrm{T}} Q_v v, \tag{32}$$

where $q_{\mathrm{ref}} = [0, 0, 0, 0]^{\mathrm{T}}$, $Q_q = \mathrm{diag}\{1, 1, 1, 1\}$, and $Q_v = \mathrm{diag}\{0.1, 0.1, 0.1, 0.1\}$. We set the stage cost of C/GMRES (FD) by

$$\begin{aligned} L(q, v, u) =& \frac{1}{2}(q - q_{\mathrm{ref}})^{\mathrm{T}} Q_q (q - q_{\mathrm{ref}}) + \frac{1}{2}v^{\mathrm{T}} Q_v v \\ &+ \frac{1}{2}u^{\mathrm{T}} R u, \end{aligned} \tag{33}$$

where $R = \mathrm{diag}\{10^{-4}, 10^{-4}, 10^{-4}, 10^{-4}\}$, and that of C/GMRES (ID) by

$$\begin{aligned} L(q, v, a, u) =& \frac{1}{2}(q - q_{\mathrm{ref}})^{\mathrm{T}} Q_q (q - q_{\mathrm{ref}}) + \frac{1}{2}v^{\mathrm{T}} Q_v v \\ &+ \frac{1}{2}a^{\mathrm{T}} Q_a a + \frac{1}{2}u^{\mathrm{T}} R u. \end{aligned} \tag{34}$$

Table 1. Average computational time [ms] per control update of C/GMRES (FD) for the four-link arm.

	$N = 10$	$N = 15$	$N = 25$	$N = 50$
$k_{\max} = 3$	0.37^*	0.55^*	0.94^*	1.8^*
$k_{\max} = 5$	0.48^*	0.75^*	1.2^*	2.4^*
$k_{\max} = 10$	8.1	1.2	2.0	4.0

* Computation diverges.

Table 2. Average computational time [ms] per control update of C/GMRES (ID) for the four-link arm.

	$N = 10$	$N = 15$	$N = 25$	$N = 50$
$k_{\max} = 3$	0.22^*	0.33^*	0.58^*	1.1^*
$k_{\max} = 5$	0.29	0.45	0.81	1.5^*
$k_{\max} = 10$	0.47	0.73	1.3	2.5

* Diagonals of Q are nonzero.

where $Q_a \in \mathbb{R}^{4\times4}$ is a diagonal matrix. Note that if we set all diagonal elements of Q_a by zero, the stage cost is the same between in C/GMRES (FD) and C/GMRES (ID). However, we set all diagonal elements of Q_a by 1.0×10^{-4} only if the numerical computation fails with setting them by 0. We simulate the swing-up control of the four-link arm using two controllers with various number of discretization of the horizon N and various iterations of the GMRES method $k_{\max}$. We choose $k_{\max}$ from $\{3, 5, 10\}$ and N from $\{10, 15, 25, 50\}$. The numerical simulations are performed under the following conditions. The initial generalized configuration of the system is $[-\pi, 0, 0, 0]^{\mathrm{T}}$ and the initial generalized velocity of the system is $[0, 0, 0, 0]^{\mathrm{T}}$. The sampling period is 1[ms] and the simulation time is 10[s]. The CPU is Intel Core i5 2.00 GHz and the C/GMRES method is written in C++. To compute ABA, RNEA, and the partial derivatives of both functions of forward and inverse dynamics, we utilize `Pinocchio` (Carpentier et al. (2019)), an efficient C++ library for the rigid-body dynamics algorithms.

Simulation Results We enumerate the average computational time per control update of C/GMRES (FD) for all $k_{\max} \in \{3, 5, 10\}$ and $N \in \{10, 15, 25, 50\}$ in Table 1 and those of C/GMRES (ID) in Table 2. Note that if the computation diverges in C/GMRES (FD), we write asterisks in Table 1. The computational time is then measured until the divergence. If the computation diverges in C/GRMRES (ID), we set the diagonal elements of Q_a 1.0×10^{-4}, which is shown as asterisks in Table 2. As a result, all the cases are succeeded in inverting the arm without the divergence. As shown in Table 1 and Table 2, the computational time of C/GMRES (ID) is less than that of C/GMRES (FD) under the same N and $k_{\max}$.

We also found that C/GMRES (ID) with setting all the diagonal elements of Q_a by zero results in little difference in control input with C/GMRES (FD). We show the simulation result of C/GMRES (ID) when $N = 25$ and $k_{\max} = 10$ in Fig. 2. C/GMRES (FD) results almost the same control as in Fig. 2 so that we cannot read the difference from the graphs like Fig. 2. This fact shows that the proposed FHOCP reduces computational time without compromising performance.

Table 3. Average computational time [ms] of C/GMRES (FD) and C/GMRES (ID) for various numbers of joints n

	$n = 8$	$n = 16$	$n = 24$	$n = 32$
C/GMRES (FD)	4.5	11	20	32
C/GMRES (ID)	2.6	6.0	10	16

4.3 Comparison of the Computational Time for Rigid-Body Systems with Various Numbers of Joints

Next, we compare the computational efficiency of the two controllers with various numbers of joints. We fix $N = 25$ and $k_{\max} = 10$ and restrict the number of joints of the controlled system n to 8, 16, 24, and 32. This subsection aims to compare the computational time of the two controllers for the control update and measure the calculation time until the calculation diverged. We list the average computational time of the control update of the two controllers in Table 3. The results show that C/GMRES (ID) reduces the computational time up to 50% from C/GMRES (FD) in every case.

5. CONCLUSIONS

In this paper, we have proposed an efficient solution method of the FHOCP for rigid body systems using RNEA and the partial derivatives of RNEA, which reduces the computational cost compared with the conventional FHOCP based on forward dynamics. We have reformulated the FHOCP with the generalized acceleration utilized as decision variables and inverse dynamics as an equality constraint. We have derived the optimality conditions and formulated the TPBVP so that it can be solved efficiently by RNEA and the partial derivatives of RNEA. Numerical experiments have shown that NMPC solving the proposed FHOCP based on inverse dynamics reduces the computational cost compared to solving the FHOCP based on forward dynamics. For future work, we will formulate the FHOCP based on inverse dynamics for under-actuated systems such as floating base systems. We will also examine the contact forces arising in realistic problems, e.g., manipulation and locomotion.

REFERENCES

Bock, H. and Plitt, K. (1984). A multiple shooting algorithm for direct solution of optimal control problems. In *9th IFAC World Congress*, 1603–1608.

Carpentier, J. and Mansard, N. (2018). Analytical derivatives of rigid body dynamics algorithms. In *Robotics: Science and Systems (RSS 2018)*, hal–01790971v2f.

Carpentier, J., Saurel, G., Buondonno, G., Mirabel, J., Lamiraux, F., Stasse, O., and Mansard, N. (2019). The Pinocchio C++ library – A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives. In *International Symposium on System Integration (SII)*, 614 – 619.

Diehl, M., Bock, H., and Schlöder, J.P. (2005). A real-time iteration scheme for nonlinear optimization in optimal feedback control. *SIAM J. Control and Optimization*, 43, 1714–1736.

Diehl, M., H. Bock, H.D., and Wieber, P.B. (2006). Fast direct multiple shooting algorithms for optimal robot

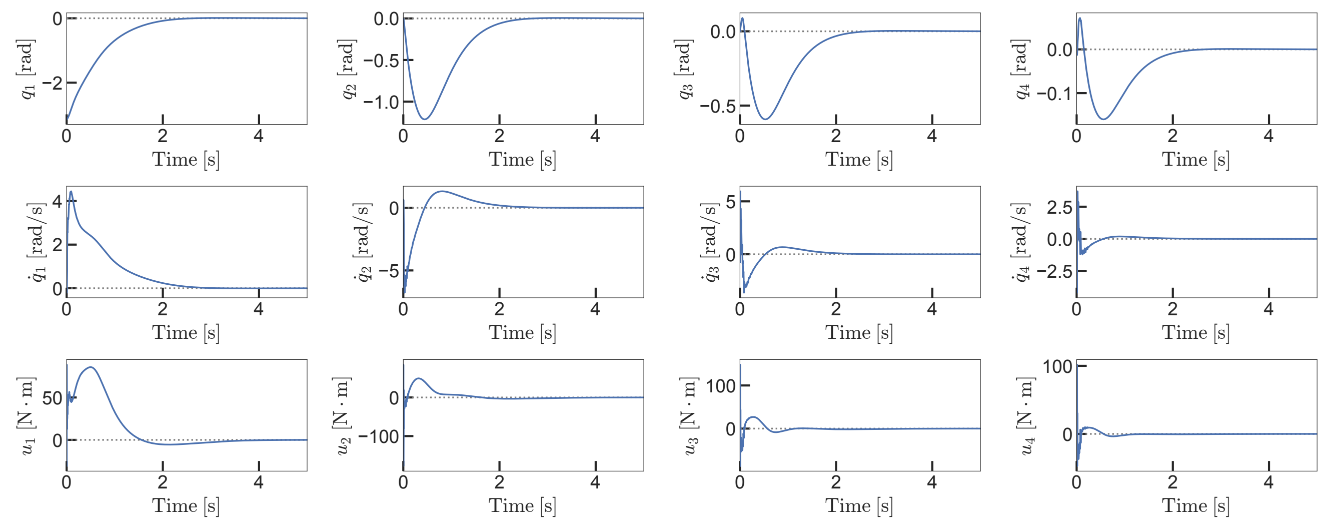

Fig. 2. Time histories of inverting the four-link arm using C/GMRES (ID) when $N = 15$ and $k_{\max} = 10$. C/GMRES (FD) results almost same time histories.

control. In M. Diehl and K. Mombaur (eds.), *Fast Motions in Biomechanics and Robotics: Optimization and Feedback Control*, volume 340 of *Lecture Notes in Control and Information Sciences*. Springer.

Erez, T. and Todorov, E. (2012). Trajectory optimization for domains with contacts using inverse dynamics. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 4914–4919.

Featherstone, R. (1983). The calculation of robot dynamics using articulated-body inertias. *The International Journal of Robotics Research*, 2(1), 13–30.

Featherstone, R. (2008). *Rigid Body Dynamics Algorithms*. Springer.

Ferreau, H.J., Kirches, C., Potschka, A., Bock, H.G., and Diehl, M. (2014). qpOASES: A parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation*, 6(4), 327–363.

Frasch, J.V., Sager, S., and Diehl, M. (2015). A parallel quadratic programming method for dynamic optimization problems. *Mathematical Programming Computation*, 7(3), 289–329.

Gerdts, M., Henrion, R., Hömberg, D., and Landry, C. (2012). Path planning and collision avoidance for robots. *Numerical Algebra, Control & Optimization*, 2(3), 437–463.

Graichen, K. and Käpernick, B. (2012). A real-time gradient method for nonlinear model predictive control. In T. Zheng (ed.), *Frontiers of Model Predictive Control*, 9 – 28.

Hsieh, C. and Liu, J. (2012). Nonlinear model predictive control for wheeled mobile robot in dynamic environment. In *2012 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, 363–368.

Kamel, M., Alexis, K., Achtelik, M., and Siegwart, R. (2015). Fast nonlinear model predictive control for multicopter attitude tracking on SO(3). In *2015 IEEE Conference on Control Applications (CCA)*, 1160–1166.

Kelly, C.T. (1995). *Iterative Methods for Linear and Nonlinear Equations*. Frontiers in Applied Mathematics. SIAM.

Koenemann, J., Del Prete, A., Tassa, Y., Todorov, E., Stasse, O., Bennewitz, M., and Mansard, N. (2015). Whole-body model-predictive control applied to the HRP-2 humanoid. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3346–3351.

Magni, L., Raimondo, D., and Allgöwer, F. (2008). *Nonlinear Model Predictive Control: Towards New Challenging Applications*. Springer.

Neunert, M., Giftthaler, M., Frigerio, M., Semini, C., and Buchli, J. (2016). Fast derivatives of rigid body dynamics for control, optimization and estimation. In *2016 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*, 91–97.

Neunert, M., Stäuble, M., Giftthaler, M., Bellicoso, C.D., Carius, J., Gehring, C., Hutter, M., and Buchli, J. (2018). Whole-body nonlinear model predictive control through contacts for quadrupeds. *IEEE Robotics and Automation Letters*, 3(3), 1458–1465.

Ohtsuka, T. (2004). A continuation/GMRES method for fast computation of nonlinear receding horizon control. *Automatica*, 40(4), 563 – 574.

Posa, M., Cantu, C., and Tedrake, R. (2014). A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research*, 33(1), 69–81.

Richter, S.L. and DeCarlo, R.A. (1983). Continuation methods: Theory and applications. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(4), 459–464.

Todorov, E. and Li, W. (2005). A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *Proceedings of the 2005, American Control Conference*, 300–306.

Zanelli, A., Domahidi, A., Jerez, J., and Morari, M. (2020). FORCES NLP: An efficient implementation of interior-point methods for multistage nonlinear nonconvex programs. *International Journal of Control*, 93(1), 13–29.