Closed-form method for the inertia-weighted input matrix utilizing O(n) forward dynamics

Gabriel Krög, Hubert Gattringer, Andreas Müller

Institute of Robotics
Johannes Kepler University Linz
Altenberger Straße 69, 4040 Linz, Austria
gabriel.kroeg@jku.at, hubert.gattringer@jku.at, a.mueller@jku.at

EXTENDED ABSTRACT

1 Introduction

In optimization-based task-space control, the inverse dynamics of the robot is often incorporated to ensure that the commanded behavior is physically feasible. For the inverse dynamics, the Equations of Motion (EoM) of the robot have to be solved in every iteration of the optimization, which slows down the optimization considerably. To avoid this, a decoupled form of the solution of the EoM can be evaluated for each time step, leaving an equation that shows a linear relation between the system inputs and its accelerations. For MBS with many degrees of freedom, efficient methods to solve the EoM become increasingly more important, which leads to various existing $O(n)$ methods, such as the Articulated-Body Algorithm by Featherstone [1] or a similar $O(n)$ method by Bremer [2], which is used in this work. Since the standard form of these methods is unable to produce mentioned decoupling of the acceleration terms, the linear map between the desired system inputs and the accelerations, called the inertia-weighted input matrix (IWIM) here, has to be derived separately. Previous work on this was done in [3], where a recursive formulation of the inverse operational-space inertia matrix was shown. For this, the contact forces were treated separately from motor torques, since this way, the properties of an inertia matrix can be retained. This paper presents an algorithm which enables the evaluation calculation of the IWIM using intermediate results obtained from an $O(n)$ method.

2 Task Control Optimization Problem

The optimization problem for task space control as shown in [4] is written as

$$\begin{aligned} \min_{\ddot{\mathbf{q}},\mathbf{u}} \; & \frac{1}{2}\left\|\mathbf{J}_t\ddot{\mathbf{q}}+\dot{\mathbf{J}}_t\dot{\mathbf{q}}-\dot{\mathbf{V}}_{t,c}\right\|^2 \\ \text{s.t. } & \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}+\mathbf{h}(\mathbf{q},\dot{\mathbf{q}})=\mathbf{B}(\mathbf{q})\mathbf{u} \end{aligned} \tag{1}$$

with the task Jacobian $\mathbf{J}_t$ giving the relationship between the task twist $\mathbf{V}_t$ and the minimal velocities $\dot{\mathbf{q}}$, $\mathbf{V}_t=\mathbf{J}_t\dot{\mathbf{q}}$. The commanded time derivative of the task twist $\dot{\mathbf{V}}_{t,c}$ contains all the necessary information about the desired task trajectory. To ensure that the resulting output is physically feasible, the EoM are considered in the constraints. As mentioned before, this problem formulation is computationally very expensive. A more efficient approach is to compute $\ddot{\mathbf{q}}=-\mathbf{M}^{-1}\mathbf{h}+\mathbf{M}^{-1}\mathbf{B}\mathbf{u}$ before solving (1). That way, the system accelerations $\ddot{\mathbf{q}}$ can be split up into two components, the free acceleration $\ddot{\mathbf{q}}_0=-\mathbf{M}^{-1}\mathbf{h}$ that only depends on the current system state and the accelerations caused by the system inputs $\ddot{\mathbf{q}}_u$, which depend linearly on the system inputs as $\ddot{\mathbf{q}}_u=\mathbf{M}^{-1}\mathbf{B}\mathbf{u}$ through the IWIM, which is $\hat{\mathbf{B}}=\mathbf{M}^{-1}\mathbf{B}$. For the derivative of the task twist, this means

$$\dot{\mathbf{V}}_t=\mathbf{J}_t\ddot{\mathbf{q}}+\dot{\mathbf{J}}_t\dot{\mathbf{q}}=\mathbf{J}_t(\ddot{\mathbf{q}}_u+\ddot{\mathbf{q}}_0)+\dot{\mathbf{J}}_t\dot{\mathbf{q}}=\mathbf{J}_t\hat{\mathbf{B}}\mathbf{u}+\mathbf{J}_t\ddot{\mathbf{q}}_0+\dot{\mathbf{J}}_t\dot{\mathbf{q}} \tag{2}$$

The IWIM is the map from the inputs to the system accelerations, which is a closely related to the cross-coupling inverse operational-space inertia matrices from [4]. Due to this separation, the nonlinear constraint that is the EoM can be directly incorporated into the cost function

$$\min_{\mathbf{u}} \frac{1}{2}\left\|\mathbf{J}_t\hat{\mathbf{B}}\mathbf{u}+\mathbf{J}_t\ddot{\mathbf{q}}_0+\dot{\mathbf{J}}_t\dot{\mathbf{q}}-\dot{\mathbf{V}}_{t,c}\right\|^2 \tag{3}$$

which speeds up the optimization considerably, as the result is an unconstrained quadratic problem and the set of optimization variables is smaller.

3 Solving the Equations of Motion

To efficiently handle the EoM of more complex systems, recursive dynamics algorithms have been proven to be the method of choice, since there are some who have a computational complexity of $O(n)$. One of them is the $O(n)$ method presented in [2], which is closely related to the Articulated-Body Algorithm (ABA) described in [1]. The idea of this algorithm is to split the system up into a set of subsystems

$$\sum_i \left(\frac{\partial \dot{\mathbf{y}}_i}{\partial \dot{\mathbf{q}}}\right)^{\top}(\mathbf{M}_i\ddot{\mathbf{y}}_i+\mathbf{h}_i(\dot{\mathbf{y}}_i,\mathbf{q},\mathbf{u}_i))=0 \tag{4}$$

and then solve the dynamics recursively on the subsystem level, thus avoiding large matrix dimensions.

Here, $\dot{\mathbf{y}}_i$ respectively $\ddot{\mathbf{y}}_i$ are a characteristic set of velocities/accelerations for subsystem i, while $\mathbf{M}_i$ and $\mathbf{h}_i$ are the corresponding mass matrix and dynamics vector. The velocities $\dot{\mathbf{y}}_i$ can be calculated recursively as $\dot{\mathbf{y}}_i = \mathbf{T}_{ip}\dot{\mathbf{y}}_p + \mathbf{F}_i\dot{\mathbf{q}}_i$ with the index p indicating the parent subsystem of i, the velocity propagation matrix $\mathbf{T}_{ip}$ from p to i and $\mathbf{F}_i$ the matrix that incorporates the free motion of the subsystem i into $\dot{\mathbf{y}}_i$. This leads to a global relationship between the set of all velocities $\dot{\mathbf{y}}_i$ and the minimal system velocities $\mathbf{F} = \frac{\partial \dot{\mathbf{y}}}{\partial \dot{\mathbf{q}}}$, which is a lower triangular block matrix. This property of $\mathbf{F}$ will be exploited later on in the new method.

The $O(n)$ method consists of three recursions. In the first one, the kinematics is evaluated from the base subsystem outwards. Then, in the following inward recursion, the articulated-body inertia $\mathbf{M}_i^*$ and bias forces $\mathbf{h}_i^*$, as they are named in the ABA, are computed as

$$\mathbf{M}_p^* = \mathbf{M}_p + \sum_{i\in\{s_j(p)\}} \mathbf{T}_{ip}^{*\top}\mathbf{M}_i^*\mathbf{T}_{ip} \qquad \mathbf{h}_p^* = \mathbf{h}_p + \sum_{i\in\{s_j(p)\}} \mathbf{T}_{ip}^{*\top}\left(\mathbf{M}_i^*\dot{\mathbf{T}}_{ip}\dot{\mathbf{y}}_p + \mathbf{h}_i^*\right), \tag{5}$$

with $\mathbf{T}_{ip}^* = (\mathbf{I} - \mathbf{M}_i^*\mathbf{F}_i\mathbf{M}_{Ri}^{-1}\mathbf{F}_i^\top)^\top\mathbf{T}_{ip}$ and $\mathbf{M}_{Ri} = \mathbf{F}_i^\top\mathbf{M}_i^*\mathbf{F}_i$. The set $\{s_j(p)\}$ denotes all subsystems that have p as its parent. Finally, the system accelerations are obtained through the final outwards recursion

$$\ddot{\mathbf{q}}_i = -\mathbf{M}_{Ri}^{-1}\mathbf{F}_i^\top\left[\mathbf{M}_i^*(\mathbf{T}_{ip}\ddot{\mathbf{y}}_p + \dot{\mathbf{T}}_{ip}\dot{\mathbf{y}}_p) + \mathbf{h}_i^*\right] \qquad \text{with} \qquad \ddot{\mathbf{y}}_i = \mathbf{T}_{ip}\ddot{\mathbf{y}}_p + \dot{\mathbf{T}}_{ip}\dot{\mathbf{y}}_p + \mathbf{F}_i\ddot{\mathbf{q}}_i + \dot{\mathbf{F}}_i\dot{\mathbf{q}}_i. \tag{6}$$

The problem here is that this method gives just the resulting accelerations without the possibility of splitting it up the way it was done in (2). In order to do that, a new formulation was developed.

4 New formulation

It is assumed that the subsystem dynamics vectors $\mathbf{h}_i$ can be split up into $\mathbf{h}_i = \mathbf{h}_{i,0} - \mathbf{B}_i\mathbf{u}_i$. Looking at the propagation formula for the bias forces in (5), a global form $\mathbf{h}_u^* = -\mathbf{B}^*\mathbf{u}$ can then be derived.

A similar separation can be done with the system accelerations. When looking at $\ddot{\mathbf{y}}_i = \frac{\partial \dot{\mathbf{y}}_i}{\partial \dot{\mathbf{q}}}\ddot{\mathbf{q}} + \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \dot{\mathbf{y}}_i}{\partial \dot{\mathbf{q}}}\right)\dot{\mathbf{q}}$ and inserting $\ddot{\mathbf{q}} = \ddot{\mathbf{q}}_0 + \ddot{\mathbf{q}}_u$, one gets $\ddot{\mathbf{y}}_i = \ddot{\mathbf{y}}_{i,0} + \left(\frac{\partial \dot{\mathbf{y}}_i}{\partial \dot{\mathbf{q}}}\right)\ddot{\mathbf{q}}_u$, which can be used to rewrite (6) to

$$\ddot{\mathbf{q}}_i = \ddot{\mathbf{q}}_{i,0} - \mathbf{M}_{Ri}^{-1}\mathbf{F}_i^\top\left[\mathbf{M}_i^*\mathbf{T}_{ip}\left(\frac{\partial \dot{\mathbf{y}}_p}{\partial \dot{\mathbf{q}}}\right)\ddot{\mathbf{q}}_u + \mathbf{h}_{i,u}^*\right]. \tag{7}$$

Extending the resulting formula for $\ddot{\mathbf{q}}_{i,u}$ to all system accelerations, the result can be simplified to

$$\ddot{\mathbf{q}}_u = \left[\mathrm{diag}(\mathbf{M}_{Ri}^{-1}\mathbf{F}_i^\top\mathbf{M}_i^*)\mathbf{F}\right]^{-1}\mathrm{diag}(\mathbf{M}_{Ri}^{-1}\mathbf{F}_i^\top)\mathbf{B}^*\mathbf{u} = \mathbf{\Gamma}^{-1}\mathrm{diag}(\mathbf{M}_{Ri}^{-1}\mathbf{F}_i^\top)\mathbf{B}^*\mathbf{u} \tag{8}$$

The matrix $\mathbf{\Gamma}$ inherits its structure from $\mathbf{F}$, i.e. it is a lower triangular block matrix. Moreover, thanks to $\mathbf{M}_{Ri}^{-1}\mathbf{F}_i^\top\mathbf{M}_i^*\mathbf{F}_i = \mathbf{I}$, all the block matrices on the diagonal are identity matrices, leading to a real lower triangular structure with only ones as its diagonal entries. This guarantees that this matrix will always be invertible.

This leaves the final result $\mathbf{J}_t\ddot{\mathbf{q}} = \mathbf{J}_t\ddot{\mathbf{q}}_0 + \mathbf{J}_t\hat{\mathbf{B}}\mathbf{u}$ with $\ddot{\mathbf{q}}_0$ obtained by running the $O(n)$ method without any inputs. Through that, all variables that are needed to calculate the global matrices $\mathbf{F}, \mathbf{B}^*$ and subsequently $\mathbf{\Gamma}$ and $\hat{\mathbf{B}}$ are computed in advance. Finally, $\hat{\mathbf{B}}$ is computed as

$$\hat{\mathbf{B}} = \left[\mathrm{diag}(\mathbf{M}_{Ri}^{-1}\mathbf{F}_i^\top\mathbf{M}_i^*)\mathbf{F}\right]^{-1}\mathrm{diag}(\mathbf{M}_{Ri}^{-1}\mathbf{F}_i^\top)\mathbf{B}^*. \tag{9}$$

A derivation of the full formulation and comparisons of the computation time will be presented in the full paper. This includes a method how $\mathbf{\Gamma}^{-1}$ can be obtained without taking an explicit matrix inverse.

Acknowledgments

This work has been supported by the "LCM - K2 Center for Symbiotic Mechatronics" within the framework of the Austrian COMET-K2 program.

References

[1] Featherstone, R.: Rigid Body Dynamics Algorithms. Springer (2008)

[2] Bremer, H.: Elastic Multibody Dynamics. Springer (2008)

[3] Wensing, P., Featherstone, R., Orin, D.E.: A reduced-order recursive algorithm for the computation of the operational-space inertia matrix. In: 2012 IEEE International Conference on Robotics and Automation. (2012)

[4] Wensing, P.M., Orin, D.E.: Generation of dynamic humanoid behaviors through task-space control with conic optimization. In: 2013 IEEE International Conference on Robotics and Automation. (2013)