Mechanism and Machine Theory 217 (2025) 106236

Contents lists available at ScienceDirect

Mechanism and Machine Theory

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

Research paper

σ-modified Lie group generalized-α methods for constrained multibody systems

Stefan Holzinger [a,*], Martin Arnold [b], Johannes Gerstmayr [c]

[a] Technical University of Munich, Chair for Applied Mechanics, Garching b. München, 85748, Bavaria, Germany
[b] Martin Luther University Halle-Wittenberg, Institute of Mathematics, Halle (Saale), 06120, Germany
[c] University of Innsbruck, Department of Mechatronics, Innsbruck, 6020, Tirol, Austria

ARTICLE INFO

Keywords:
Generalized-α
Newmark
Lie group time integration
Constrained flexible multibody systems
Algorithmic dissipation

ABSTRACT

Efficient and accurate time integration methods are crucial for real-time simulation, optimization and control of constrained multibody systems. This paper presents new Lie group generalized-α methods that improve accuracy for multibody systems with large rotations. The proposed methods extend the widely used *geom1* scheme by Brüls and Cardona by introducing a σ-modification that allows to systematically eliminate a Lie group-specific part of the leading error term without compromising second-order accuracy or zero stability. While optimal accuracy is achieved for a specific choice of σ, the special case $\sigma = 1$ offers notable algorithmic simplicity and minimal computational overhead. The original *geom1* scheme is recovered by setting $\sigma = 0$. Several numerical benchmarks demonstrate the potential of the proposed Lie group integrators compared to both the original *geom1* method and conventional formulations based on Euler parameters or Cardan/Tait-Bryan angles.

1. Introduction

Efficient and accurate time integration methods are indispensable for real-time applications, large-scale simulations, parametric studies, or optimization of constrained multibody systems. The generalized-α method [1] is a Newmark-type integrator that has become a standard method for time integration of constrained multibody systems. The method is second-order accurate, easy to implement, computationally efficient and allows an optimal combination of accuracy at low-frequency and numerical damping at high-frequency [2]. It is also well-suited for systems involving controller dynamics and supports implementations with variable time step sizes, cf. [3]. It can be combined easily with index reduction techniques [4,5] and extensions for non-smooth systems exist [6,7], making it a powerful time integration method for the mentioned applications, see e.g. [3]. This paper introduces a new variant of the generalized-α method that not only provides improved accuracy for multibody systems with large rotations, but also allows a remarkably simple implementation, making it well suited for real-time applications, model-based control, and optimization.

A widely used approach for such applications – also adopted in this paper – is to model multibody systems using absolute coordinates [8]. That is, a set of absolute nodal translation and rotation variables describing the position and orientation of each body relative to an inertial frame, see e.g. [9]. Each translation variable resides in the linear space $\mathbb{R}^3$ whereas each rotation variable belongs to special orthogonal group SO(3) or an isomorphic representation[1] of it. Classical parameterization based approaches explicitly represent rotation variables in a coordinate system, which allows to formulate the equations of motion in a linear space

* Corresponding author.
E-mail addresses: stefan.holzinger@tum.de (S. Holzinger), martin.arnold@mathematik.uni-halle.de (M. Arnold), johannes.gerstmayr@uibk.ac.at (J. Gerstmayr).

[1] i.e., rotation parameters as Euler parameters or the Cartesian rotation vector for example.

https://doi.org/10.1016/j.mechmachtheory.2025.106236
Received 29 August 2025; Accepted 27 September 2025

Available online 6 October 2025

instead of a nonlinear one. Consequently, standard integrators like generalized-α [1] can be used for time integration. A well-known issue in the time integration of the resulting equations of motion are singular points, which arise when spatial rotations are modeled using three rotation parameters [8]. In classical parameterization based approaches, singular points are typically avoided by employing reparameterization strategies [9,10] or by using four Euler parameters in combination with differential–algebraic equations (DAEs), see e.g. [11].

It is well-known that significant simplifications and improved computational performance can be obtained for an absolute coordinate approach, if the equations of motion as well as the integration algorithm are directly formulated in a Lie group setting, see e.g. [12,13]. Lie group integrators address the challenges in time integration of spatial rotations by algorithms that exploit the Lie group structure of the configuration spaces, see e.g. [14]. The configuration space of a single frame or rigid body is given by the direct or semi-direct product of $\mathbb{R}^3$ and SO(3) [9]. Rodrigues like formulae allow the efficient evaluation of the exponential map and its derivatives that are essential algorithmic components of Lie group integrators, see, e.g., [15–17]. At system level, the configuration space is composed as tensor product of its components representing the individual frames and rigid bodies. The efficient implementation of Lie group integrators for multibody systems relies on these specific properties of configuration spaces with Lie group structure.

Park and Chung [16] combined local coordinates on the Lie group with an explicit one-step Lie group integrator of Munthe-Kaas-type [15], resulting in a method that achieves the same order as its classical counterpart from ODE theory in linear spaces. In each stage of the method, they use a Rodrigues-like formula for evaluating the inverse of the tangent operator that represents the derivative of the exponential map. For multistep methods [18] and for implicit integrators [19], the use of local parametrizations in a Munthe-Kaas-like framework is less straightforward and algorithmically challenging.

As an alternative, Brüls and Cardona [20] proposed a novel generalized-α Lie group integrator that systematically avoids all forms of nested iterations while still achieving the same order as its classical counterpart [12]. A detailed comparison with a local coordinate-based updated Lagrangian approach [19] is given in sections 6 and 7 of [12]. With local coordinates, the method is algorithmically substantially more complex but achieves in most numerical tests a better accuracy, see [12].

The critical view of Mäkinen [21] on Newmark-type Lie group integrators, along with growing interest in implementation aspects [13], motivates the present investigation into local coordinate based approaches for generalized-α Lie group time integration methods. In this paper, we present novel local coordinates based Lie group generalized-α methods for constrained multibody systems. The newly presented σ-modified Lie group generalized-α methods are modifications of the geom1 method of Brüls and Cardona [20], which is the most frequently used generalized-α Lie group integrator in multibody dynamics, see [12]. As will be shown in this paper, the σ-modification – unlike geom1 – allows to eliminate a Lie group specific part of the leading error term without compromising the method's second-order accuracy or zero stability. Furthermore, we demonstrate that the σ-modification allows for the systematic removal of the tangent operator from the algorithm, resulting in a simple and computationally efficient algorithm.

The rest of the paper is organized as follows: In Section 2, we revisit the equations of motion of constrained multibody systems formulated within the Lie group framework. Subsequently, we address local coordinates in Section 3. The new σ-modified Lie group generalized-α methods are introduced in Section 4. In Section 5, the convergence of the method is studied in detail and second-order accuracy is proven. Implementation aspects are discussed in detail in Section 6. In Section 7, we compare our σ-modified Lie group generalized-α methods with the geom1 method and a conventional Euler parameter based formulation in terms of accuracy and computational performance using several numerical examples. Finally, we conclude our study in Section 8.

2. Equations of motion on a Lie group

This section provides a recap of the equations of motion for constrained multibody systems as formulated in the Lie group setting.

The dynamics of multibody systems are effectively modeled by DAEs defined on a k-dimensional manifold G with Lie group structure [12]. In this paper, we consider constrained multibody systems of the form

$$\mathbf{M}(q)\dot{\mathbf{v}} = -\mathbf{g}(q,\mathbf{v},t) - \mathbf{B}^{\mathrm{T}}(q)\boldsymbol{\lambda}\ , \tag{1}$$

$$\dot{q} = DL_q(e)\cdot\widetilde{\mathbf{v}}\ , \tag{2}$$

$$\boldsymbol{\Phi}(q) = \mathbf{0}\ , \tag{3}$$

$$\text{and/or}\quad \dot{\boldsymbol{\Phi}}(q,\mathbf{v}) = \mathbf{0}\ . \tag{4}$$

In an absolute coordinate approach, an element $q \in G$ comprises multiple subsets of absolute nodal displacement and rotation variables, initially treated as independent. Generally, these variables must satisfy a set of m kinematic constraints at the position $\boldsymbol{\Phi} \in \mathbb{R}^m$ or velocity level $\dot{\boldsymbol{\Phi}} \in \mathbb{R}^m$, restricting the dynamics to a submanifold of dimension $k-m$ [12]. Matrix $\widetilde{\mathbf{v}}$ denotes the skew symmetric representation of the velocity vector $\mathbf{v} \in \mathbb{R}^k$, and $\boldsymbol{\lambda} \in \mathbb{R}^m$ is the vector of Lagrange multipliers associated with either the constraints on position or velocity level. The invertible linear mapping

$$\widetilde{(\bullet)} : \mathbb{R}^k \to \mathfrak{g}\ , \quad \mathbf{w} \mapsto \widetilde{\mathbf{w}}\ , \tag{5}$$

defines an isomorphism between vectors in $\mathbb{R}^k$ and the Lie algebra $\mathfrak{g} := T_eG$, which is defined as the tangent space at the neutral element $e \in G$ [14]. The mass matrix $\mathbf{M}(q) \in \mathbb{R}^{k\times k}$ is supposed to be symmetric and positive definite. Vector $\mathbf{g} \in \mathbb{R}^k$ represents external, internal, and complementary inertia forces, and $\mathbf{B} \in \mathbb{R}^{m\times k}$ is the matrix of constraint gradients

$$\mathbf{B}(q) = \frac{\partial \dot{\boldsymbol{\Phi}}(q,\mathbf{v})}{\partial \mathbf{v}}\ . \tag{6}$$

The kinematic reconstruction Eq. (2) relates velocities $\mathbf{v}$ and the time derivative of the configuration variable q through the directional derivative $DL_q(e)$ of the left translation map $L_q(e)$, cf. [12,22]. Eq. (2) is typically the starting point for Lie group time integration methods in multibody dynamics and is discussed in greater detail in the next section.

3. Local coordinates and velocities

The proposed Lie group generalized-α methods presented later in Section 4 are formulated using local coordinates. This section explains how the time derivatives of local coordinates relate to velocities and how these are used to integrate the kinematic reconstruction equation Eq. (2).

Let Ψ be an invertible map that maps a neighborhood of $\widetilde{\mathbf{0}} \in \mathfrak{g}$ in the Lie algebra to the Lie group

$$\Psi : \mathfrak{g} \to G, \quad \tilde{\boldsymbol{\theta}} \mapsto q = \Psi(\tilde{\boldsymbol{\theta}}) , \tag{7}$$

such that $\tilde{\mathbf{0}} \in \mathfrak{g}$ is mapped to the identity of G. A solution to Eq. (2) at time $t \in [t_n, t_{n+1}]$ with initial conditions $q(t_n) \in G$ is given by

$$q(t) = q(t_n) \circ \Psi(\tilde{\boldsymbol{\theta}}(t)) \in G , \tag{8}$$

where $\tilde{\boldsymbol{\theta}}(t)$ is the solution of the ODE

$$\widetilde{\mathbf{v}} = \mathrm{d}\Psi_{-\tilde{\boldsymbol{\theta}}}(\dot{\tilde{\boldsymbol{\theta}}}) \quad \overset{\widetilde{(\cdot)}}{\longleftrightarrow} \quad \mathbf{v} = \mathbf{T}_\Psi(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} \tag{9}$$

with initial conditions $\boldsymbol{\theta}(t_n) = \mathbf{0}$, cf. [8,20]. The vector $\boldsymbol{\theta}$ is the vector of local coordinates on G, which locally parametrizes the configuration increment from time step n to time step $n+1$ [8]. The symbol $\circ$ in Eq. (8) denotes the composition operation for group elements, see e.g. [22]. Matrix $\mathrm{d}\Psi_{-\tilde{\boldsymbol{\theta}}}$ is the left-trivialized differential of Ψ, see e.g. [8]. The matrix exponential

$$\exp(\tilde{\boldsymbol{\theta}}) = \sum_{i=0}^{\infty} \frac{1}{i!}\tilde{\boldsymbol{\theta}}^i \tag{10}$$

is a typical choice for a coordinate map [8] and is used in this paper. Efficient closed-form solutions to Eq. (10) exist for the Lie groups $\mathbb{R}^3 \times SO(3)$ and $SE(3)$; see [14,22,23]. The matrix $\mathbf{T}_\Psi$ in Eq. (9) is denoted as tangent operator [20] and its specific form depends on the considered Lie group and the chosen coordinate map. Closed-form expressions for the tangent operator can be found, for instance, in [24,25].

4. Proposed σ-modified Lie group generalized-α methods

Inspired by the works [12,14,19,20,26], we introduce in this section the proposed σ-modified Lie group generalized-α methods for solving Eqs. (1)–(4). In the proposed Lie group generalized-α methods, the numerical solution is updated in time step $t \to t+h$ with time step size h based on the following discretized set of equations

$$q_{n+1} = q_n \circ \exp(\tilde{\boldsymbol{\theta}}_{n+1}) , \tag{11}$$

$$\boldsymbol{\theta}_{n+1} = h\mathbf{v}_n + h\dot{\boldsymbol{\theta}}^{(\sigma)}_{n+1} + h^2(\tfrac{1}{2} - \beta)\mathbf{a}_n + h^2\beta\mathbf{a}_{n+1} , \tag{12}$$

$$\dot{\boldsymbol{\theta}}^{(\sigma)}_{n+1} = \sigma\frac{\beta}{\gamma}(\dot{\boldsymbol{\theta}}_{n+1} - \mathbf{v}_{n+1}) - \mathbf{T}^{-1}(\boldsymbol{\theta}_{n+1})\mathbf{B}^{\mathrm{T}}(q_n)\boldsymbol{\eta}_n , \tag{13}$$

$$\mathbf{v}_{n+1} = \mathbf{v}_n + h\,(1-\gamma)\,\mathbf{a}_n + h\gamma\mathbf{a}_{n+1} , \tag{14}$$

$$(1-\alpha_m)\mathbf{a}_{n+1} + \alpha_m\mathbf{a}_n = (1-\alpha_f)\dot{\mathbf{v}}_{n+1} + \alpha_f\dot{\mathbf{v}}_n , \tag{15}$$

$$\mathbf{T}(\boldsymbol{\theta}_{n+1})\dot{\boldsymbol{\theta}}_{n+1} = \mathbf{v}_{n+1} , \tag{16}$$

and satisfies the equilibrium conditions

$$\mathbf{M}(q_{n+1})\dot{\mathbf{v}}_{n+1} = -\mathbf{g}(q_{n+1}, \mathbf{v}_{n+1}, t_{n+1}) - \mathbf{B}^{\mathrm{T}}(q_{n+1})\boldsymbol{\lambda}_{n+1} , \tag{17}$$

$$\boldsymbol{\Phi}(q_{n+1}) = \mathbf{0} , \tag{18}$$

$$\text{and/or} \quad \dot{\boldsymbol{\Phi}}(q_{n+1}, \mathbf{v}_{n+1}) = \mathbf{0} , \tag{19}$$

$$\text{and} \quad \boldsymbol{\eta}_n = \mathbf{0} \quad \text{(index-3 and index 2 formulation).} \tag{20}$$

As compared to the works [12,14,20], the proposed generalized-α methods add a new term $\dot{\boldsymbol{\theta}}^{(\sigma)}_{n+1}$ with a new algorithmic parameter $\sigma \in \mathbb{R}$ to the integration formula Eq. (12). The variable $\dot{\boldsymbol{\theta}}^{(\sigma)}_{n+1}$ consistently incorporates the kinematic relation between the time derivative of local coordinates $\dot{\boldsymbol{\theta}}$ and velocities $\mathbf{v}$ according to Eq. (9) at time step $n+1$. For the index-3 formulation, $\sigma = 0$ yields the original `geom1` method of Brüls and Cardona [20]. As we will show in Section 5, optimal accuracy is obtained for $\sigma = \gamma/(3\beta)$. For $\sigma = 1$, a particular simple and implementation friendly integrator is obtained, cf. Section 6.3. The four parameters β, γ, α_f and α_m in Eqs. (12)–(15) represent the usual generalized-α parameters and should be selected as usual in order to obtain suitable convergence and stability properties, see e.g. [2].

Note, the algorithmic acceleration variable $\mathbf{a}$ is associated with the accelerations $\dot{\mathbf{v}}$, see Eq. (15). Time derivatives of local coordinates and accelerations are related by [12,19]

$$\dot{\mathbf{v}} = \mathbf{T}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \dot{\mathbf{T}}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} \ . \tag{21}$$

As $\ddot{\boldsymbol{\theta}} \neq \dot{\mathbf{v}}$ in general, the algorithmic acceleration variables $\mathbf{a}_n$ and $\mathbf{a}_{n+1}$ in Eq. (12) are not consistent to Eq. (21). However, we will prove in Section 5 that the proposed Lie group generalized-α methods remain second-order accurate and can give improved accuracy as compared to the original geom1 method.

5. Convergence

For local truncation error analysis, the discretized equations of motion are evaluated in terms of the analytical solution $q(t)$, $\mathbf{v}(t)$, $\dot{\mathbf{v}}(t)$, $\lambda(t)$ with residuals that are studied for $h \to 0$ by Taylor expansion, see section 4.1 of [14] for some Lie group specific details. The term $-\mathbf{B}^{\mathrm{T}}(q_n)\boldsymbol{\eta}_n$ in (13) may be ignored for the moment since $\boldsymbol{\eta}(t) \equiv \mathbf{0}$ for the analytical solution. As in [27], the algorithmic acceleration variables $\mathbf{a}_n$, $\mathbf{a}_{n+1}$ are substituted by $\dot{\mathbf{v}}(t_n + \Delta_\alpha h)$ and $\dot{\mathbf{v}}(t_{n+1} + \Delta_\alpha h)$ with the shift parameter $\Delta_\alpha := \alpha_m - \alpha_f$. Furthermore, the algorithmic parameters have to satisfy the order condition $\gamma = \frac{1}{2} - \Delta_\alpha$, see [1].

In local coordinates $\boldsymbol{\theta}$ with Ψ being defined by the exponential map exp, the solution at $t_{n+1} = t_n + h$ is given by

$$q(t_n + h) = q(t_n) \circ \exp\big(\tilde{\boldsymbol{\theta}}(t_n + h)\big) \ , \tag{22}$$

see (8), with [28]

$$\boldsymbol{\theta}(t_n + h) = h\mathbf{v}(t_n) + \frac{h^2}{2}\dot{\mathbf{v}}(t_n) + \frac{h^3}{6}\ddot{\mathbf{v}}(t_n) + \frac{h^3}{12}\,\widehat{\mathbf{v}}(t_n)\dot{\mathbf{v}}(t_n) + \mathcal{O}(h^4) \ . \tag{23}$$

Here, we have used the hat operator $\widehat{(\bullet)} : \mathbb{R}^k \times \mathbb{R}^{k\times k}$ that represents as in [22] the adjoint operator (matrix commutator) in the sense of

$$\widetilde{\widehat{\mathbf{v}}_a\mathbf{v}_b} = \mathrm{Ad}_{\tilde{\mathbf{v}}_a}(\tilde{\mathbf{v}}_b) = \big[\tilde{\mathbf{v}}_a, \tilde{\mathbf{v}}_b\big] \ . \tag{24}$$

Adjoint operator and matrix commutator account for the non-commutativity of group operation and matrix multiplication, respectively. They may be used to represent the tangent operator $\mathbf{T}(\mathbf{v}_a)$ by its series expansion [15]

$$\mathbf{T}(\mathbf{v}_a) = \sum_{i=0}^{\infty} \frac{(-1)^i}{(i+1)!}(\widehat{\mathbf{v}}_a)^i \tag{25}$$

that proves $\mathbf{T}(\mathbf{v}_a)\mathbf{v}_b = \mathbf{v}_b$ for vectors $\mathbf{v}_a$, $\mathbf{v}_b$ corresponding to commuting elements $\tilde{\mathbf{v}}_a, \tilde{\mathbf{v}}_b \in \mathfrak{g}$ since $\big[\tilde{\mathbf{v}}_a, \tilde{\mathbf{v}}_b\big] = \tilde{\mathbf{0}}$ in that case, i.e., $(\widehat{\mathbf{v}}_a)^i\,\mathbf{v}_b = \mathbf{0}$, $(i > 0)$. In particular, we have $\mathbf{T}(\boldsymbol{\theta}_{n+1})\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_{n+1}$ resulting in

$$\big(\big(\mathbf{T}(\boldsymbol{\theta}_{n+1})\big)^{-1} - \mathbf{I}\big)\cdot \tfrac{1}{h}\boldsymbol{\theta}_{n+1} = \mathbf{0} \ . \tag{26}$$

This identity helps to simplify the convergence analysis by expressing the vector $h\dot{\boldsymbol{\theta}}^{(\sigma)}_{n+1}$ in the right hand side of Eq. (12) formally in terms of $\mathbf{v}_n$, $\mathbf{a}_n$, $\mathbf{a}_{n+1}$. To this end, we omit again the $-\mathbf{B}^{\mathrm{T}}(q_n)\boldsymbol{\eta}_n$ term in (13) and obtain $\dot{\boldsymbol{\theta}}_{n+1} = \big(\mathbf{T}(\boldsymbol{\theta}_{n+1})\big)^{-1}\mathbf{v}_{n+1}$, i.e.

$$\dot{\boldsymbol{\theta}}^{(\sigma)}_{n+1} = \sigma\tfrac{\beta}{\gamma}\big(\big(\mathbf{T}(\boldsymbol{\theta}_{n+1})\big)^{-1} - \mathbf{I}\big)\,\mathbf{v}_{n+1} \ , \tag{27}$$

$$= \sigma\tfrac{\beta}{\gamma}\big(\big(\mathbf{T}(\boldsymbol{\theta}_{n+1})\big)^{-1} - \mathbf{I}\big)\big(\mathbf{v}_{n+1} - \tfrac{1}{h}\boldsymbol{\theta}_{n+1}\big) \ , \tag{28}$$

see (13) and (26). The inverse of the tangent operator in Eq. (28) satisfies

$$\big(\mathbf{T}(h\mathbf{v})\big)^{-1} = \mathbf{I} + \tfrac{h}{2}\,\widehat{v} + \mathcal{O}(h^2) \ , \tag{29}$$

since

$$\mathbf{T}(h\mathbf{v}) = \mathbf{I} - \tfrac{h}{2}\,\widehat{v} + \mathcal{O}(h^2) \tag{30}$$

according to Eq. (25). Therefore,

$$\big(\mathbf{T}(\boldsymbol{\theta}_{n+1})\big)^{-1} = \big(\mathbf{T}(h\mathbf{v}_n + h\dot{\boldsymbol{\theta}}^{(\sigma)}_{n+1} + h^2(\tfrac{1}{2} - \beta)\mathbf{a}_n + h^2\beta\mathbf{a}_{n+1})\big)^{-1} \ , \tag{31}$$

$$= \mathbf{I} + \tfrac{h}{2}\,\widehat{\mathbf{v}}_n + \tfrac{h}{2}\,\widehat{\dot{\boldsymbol{\theta}}}^{(\sigma)}_{n+1} + \tfrac{h^2}{2}\,\mathbf{J}(\mathbf{v}_n, \dot{\boldsymbol{\theta}}^{(\sigma)}_{n+1}, \mathbf{a}_n, \mathbf{a}_{n+1}) \tag{32}$$

with a matrix-valued function $\mathbf{J}$ that depends smoothly on all its arguments. Inserting Eqs. (12), (14), (32) in Eq. (28), we obtain

$$\dot{\boldsymbol{\theta}}^{(\sigma)}_{n+1} = \tfrac{h}{2}\,\sigma\tfrac{\beta}{\gamma}\big(\widehat{\mathbf{v}}_n + \widehat{\dot{\boldsymbol{\theta}}}^{(\sigma)}_{n+1} + h\,\mathbf{J}\big)\big(\tfrac{h}{2}\mathbf{a}_n + h(\gamma - \beta)(\mathbf{a}_{n+1} - \mathbf{a}_n) - \dot{\boldsymbol{\theta}}^{(\sigma)}_{n+1}\big) \ . \tag{33}$$

The Implicit function theorem allows to solve these equations locally uniquely with respect to $\dot{\boldsymbol{\theta}}^{(\sigma)}_{n+1}$ whenever the coefficient $\frac{h}{2} > 0$ in front of the right hand side is sufficiently small:

$$\dot{\boldsymbol{\theta}}^{(\sigma)}_{n+1} = \sigma\,h^2\boldsymbol{\varepsilon}^{(\sigma)}(\mathbf{v}_n, \mathbf{a}_n, \mathbf{a}_{n+1}) \tag{34}$$

with a locally uniquely defined function $\boldsymbol{\varepsilon}^{(\sigma)}$ that satisfies a Lipschitz condition with respect to all three arguments $\mathbf{v}_n$, $\mathbf{a}_n$, $\mathbf{a}_{n+1}$ (with Lipschitz constants being independent of h). For $h \to 0$, the term $h^2\boldsymbol{\varepsilon}^{(\sigma)}$ is given by

$$h^2\boldsymbol{\varepsilon}^{(\sigma)}(\mathbf{v}_n,\mathbf{a}_n,\mathbf{a}_{n+1}) = \frac{h}{2}\,\frac{\beta}{\gamma}\big(\frac{h}{2}\,\widehat{\mathbf{v}}_n\mathbf{a}_n + h(\gamma-\beta)\widehat{\mathbf{v}}_n(\mathbf{a}_{n+1}-\mathbf{a}_n)\big) + \mathcal{O}(h^3)\,. \tag{35}$$

Note, that the formal expression (34) for $\dot{\boldsymbol{\theta}}^{(\sigma)}_{n+1}$ is just introduced as a tool for the convergence analysis. It is not at all relevant for the practical implementation of the σ-modified method that will be discussed in Section 6 below.

With (34), the discretized Eq. (12) with $\sigma \neq 0$ is seen to be an $\mathcal{O}(h^3)$ perturbation of the corresponding equation for the original geom1 method of Brüls and Cardona [20]:

$$\boldsymbol{\theta}_{n+1} = h\mathbf{v}_n + h^2(\tfrac{1}{2}-\beta)\mathbf{a}_n + h^2\beta\mathbf{a}_{n+1} + \sigma\, h^3\boldsymbol{\varepsilon}^{(\sigma)}(\mathbf{v}_n,\mathbf{a}_n,\mathbf{a}_{n+1})\,. \tag{36}$$

The new $\sigma\, h^3\boldsymbol{\varepsilon}^{(\sigma)}$ term in (36) contributes additional higher order terms to the theoretical investigations that do, however, not affect the (quite complex) analysis of zero stability and (global) error propagation for generalized-α Lie group integrators in section 4 of [14]. This comment applies as well to the stabilized index-2 version of the σ-modified method with the additional $-\mathbf{B}^{\mathrm{T}}(q_n)\boldsymbol{\eta}_n$ term in Eq. (13), see Lemma 4.9 in [14] for a more detailed discussion.

For local truncation error analysis, vectors $\mathbf{v}_n$, $\mathbf{a}_n$, $\mathbf{a}_{n+1}$ in (36) have to be substituted by $\mathbf{v}(t_n)$, $\dot{\mathbf{v}}(t_n+\Delta_\alpha h)$ and $\dot{\mathbf{v}}(t_{n+1}+\Delta_\alpha h)$, respectively, resulting in a vector

$$\boldsymbol{\theta}_{n+1}(t_n+h) := h\mathbf{v}(t_n) + h^2(\tfrac{1}{2}-\beta)\dot{\mathbf{v}}(t_n+\Delta_\alpha h) + h^2\beta\dot{\mathbf{v}}(t_{n+1}+\Delta_\alpha h) + \sigma\, h^3\boldsymbol{\varepsilon}^{(\sigma)}_h(t_n) \tag{37}$$

with

$$\boldsymbol{\varepsilon}^{(\sigma)}_h(t_n) := \boldsymbol{\varepsilon}^{(\sigma)}\big(\mathbf{v}(t_n), \dot{\mathbf{v}}(t_n+\Delta_\alpha h), \dot{\mathbf{v}}(t_{n+1}+\Delta_\alpha h)\big) \tag{38}$$

that is compared to the analytical solution $\boldsymbol{\theta}(t_n+h)$ in Eq. (23).

The σ-modification contributes a new error term of size $\mathcal{O}(h^3)$ that does not affect the (known) second order local error estimate of the geom1 method but yields a different leading error term. More precisely, Eqs. (35) and (38) show

$$h^3\boldsymbol{\varepsilon}^{(\sigma)}_h(t_n) = \frac{h^3}{4}\,\frac{\beta}{\gamma}\;\widehat{\mathbf{v}}(t_n)\dot{\mathbf{v}}(t_n) + \mathcal{O}(h^4) \tag{39}$$

with the Lie group specific third order term $\widehat{\mathbf{v}}(t_n)\dot{\mathbf{v}}(t_n)$ that is known from the Taylor expansion of the analytical solution in Eq. (23). Note, that for $\sigma = 0$, i.e., for the original geom1 method, this term does not appear in the Taylor expansion of $\boldsymbol{\theta}_{n+1}(t_n+h)$, see (37). The proposed σ-modification offers a systematic way to address this Lie group specific third order term in the difference of $\boldsymbol{\theta}(t_n+h)$ and $\boldsymbol{\theta}_{n+1}(t_n+h)$ and therefore also in the local truncation error $\mathbf{l}^q_n$ for the configuration variable q.

Following the analysis of Lemma 4.2 in [14], we get a local truncation error

$$\mathbf{l}^q_n = C_q\, h^3\,\ddot{\mathbf{v}}(t_n) + \frac{h^3}{4}\,(\tfrac{1}{3}-\sigma\tfrac{\beta}{\gamma})\widehat{\mathbf{v}}(t_n)\dot{\mathbf{v}}(t_n) + \mathcal{O}(h^4) \tag{40}$$

with

$$C_q = \tfrac{1}{6}(1-6\beta-3\Delta_\alpha)\,. \tag{41}$$

For the algorithmically most promising setting $\sigma = 1$, see Section 6.3 below, the classical algorithmic parameters according to Chung and Hulbert [1] result in

$$\frac{1}{3} - \sigma\frac{\beta}{\gamma} = \frac{1}{3} - \frac{\beta}{\gamma} = \frac{1}{3}\Big(1 - \frac{2}{(1+\rho_\infty)(1-\rho_\infty/3)}\Big) \tag{42}$$

with ρ_∞ denoting the numerical damping parameter. A straightforward analysis proves $|\frac{1}{3}-\frac{\beta}{\gamma}| < \frac{1}{3}$ whenever $\rho_\infty \in (0,1]$. I.e., the coefficient of the Lie group specific term $\widehat{\mathbf{v}}(t_n)\dot{\mathbf{v}}(t_n)$ in (40) is in modulus systematically reduced by the σ-modification with $\sigma = 1$. The term may even be completely eliminated from the leading error term defining the new algorithmic parameter σ such that $\sigma\frac{\beta}{\gamma} = \frac{1}{3}$.

The results of the convergence analysis may be summarized as follows: σ-modification does not affect the order of the Lie group integrator but allows to reduce or even to eliminate a Lie group specific part in the leading error term. The σ-modified method shares the favorable zero-stability properties of the classical geom1 integrator for unconstrained systems and for constrained systems in index-3, index-2 or stabilized index-2 formulation. With appropriate starting values, the method converges with order $p = 2$ in all solution components. Less sophisticated initialization schemes may result (as in geom1) in a transient first order error term that is, however, damped out rapidly by numerical dissipation [14].

6. Implementation

In this section, we address in detail implementation aspects of our σ-modified Lie group generalized-α methods and present Newton–Raphson algorithms that solve Eqs. (11)–(19) at time step $n+1$ for the variables q_{n+1}, $\mathbf{v}_{n+1}$, $\dot{\mathbf{v}}_{n+1}$, $\mathbf{a}_{n+1}$, λ_{n+1}, starting from given values q_n, $\mathbf{v}_n$, $\dot{\mathbf{v}}_n$, $\mathbf{a}_n$.

6.1. Local-global-transition map

For efficient implementation, we do not operate with matrix representations of Lie group elements $q \in G$. Instead, we use an isomorphic representation in terms of absolute coordinates $\mathbf{q} \in \mathbb{R}^k$, i.e.,

$$q \cong \mathbf{q} \, . \tag{43}$$

Eq. (8) can be replaced by the so-called Local-Global-Transition (LGT) map $\boldsymbol{\tau}$ [8], which maps local coordinates $\boldsymbol{\theta}(t)$ and initial values $\mathbf{q}(t_n)$ to absolute coordinates $\mathbf{q}(t_{n+1})$

$$\begin{aligned} \boldsymbol{\tau} : \mathbb{R}^k \times \mathfrak{g} &\to \mathbb{R}^k \, , \\ (\mathbf{q}(t_n), \boldsymbol{\theta}(t)) &\mapsto \mathbf{q}(t) = \boldsymbol{\tau}(\mathbf{q}(t_n), \boldsymbol{\theta}(t)) \, . \end{aligned} \tag{44}$$

The explicit form of the LGT map depends on the selected type of absolute coordinates and the chosen Lie group formulation,[2] as discussed in [8]. LGT maps used in this paper are provided in Appendix A.

6.2. Newton–Raphson

In this section, we present a Newton–Raphson algorithm for solving the index-3 system Eqs. (1)–(4). Newton–Raphson algorithms for the index-2 and stabilized index-2 formulation can be derived in the same way from Eqs. (11)–(16) and are not addressed here due to space reasons. For completeness, Appendix D summarizes the key formulas required for implementing the proposed stabilized index-2 integrator.

We define the residual vector $\mathbf{r}$ as

$$\mathbf{r} := \mathbf{M}(\mathbf{q})\dot{\mathbf{v}} + \mathbf{g}(\mathbf{q}, \mathbf{v}, t) + \mathbf{B}^{\mathrm{T}}(\mathbf{q})\lambda = \mathbf{0} \, . \tag{45}$$

The linearization of Eq. (45) and Eqs. (18)–(19) follows as

$$\Delta \mathbf{r} = \mathbf{M}\Delta\dot{\mathbf{v}}_{n+1} + \mathbf{C}_t \Delta \mathbf{v}_{n+1} + \mathbf{K}_t \Delta \mathbf{q}_{n+1} + \mathbf{B}^{\mathrm{T}} \Delta \lambda_{n+1} \, , \tag{46}$$

$$\Delta \boldsymbol{\Phi} = \mathbf{B}\Delta \mathbf{q}_{n+1} \, , \tag{47}$$

$$\Delta \dot{\boldsymbol{\Phi}} = \mathbf{Z}_t \Delta \mathbf{q}_{n+1} + \mathbf{B}\Delta \mathbf{v}_{n+1} \, , \tag{48}$$

where

$$\mathbf{K}_t(\mathbf{q}, \mathbf{v}, \dot{\mathbf{v}}, \lambda, t) = \left. \frac{\partial \mathbf{r}(\boldsymbol{\tau}(\mathbf{q}, \mathbf{z}), \mathbf{v}, \dot{\mathbf{v}}, \lambda, t)}{\partial \mathbf{z}} \right|_{\mathbf{z}=\mathbf{0}} , \tag{49}$$

$$\mathbf{C}_t(\mathbf{q}, \mathbf{v}, t) = \left. \frac{\partial \mathbf{r}(\mathbf{q}, \mathbf{z}, t)}{\partial \mathbf{z}} \right|_{\mathbf{z}=\mathbf{v}} , \tag{50}$$

$$\mathbf{Z}_t(\mathbf{q}, \mathbf{v}) = \left. \frac{\partial \dot{\boldsymbol{\Phi}}(\boldsymbol{\tau}(\mathbf{q}, \mathbf{z}) \, , \mathbf{v})}{\partial \mathbf{z}} \right|_{\mathbf{z}=\mathbf{0}} . \tag{51}$$

In the practical implementation of the Newton–Raphson algorithm, we neglect the terms $(\partial \mathbf{M}(\mathbf{q})/\partial \mathbf{q})\Delta \mathbf{q}$ and $(\partial(\mathbf{B}^{\mathrm{T}}(\mathbf{q})\lambda)/\partial \mathbf{q})\Delta \mathbf{q}$. We linearize the integration formulas Eqs. (11)–(16) as

$$\Delta \mathbf{q}_{n+1} = \mathbf{T}(\boldsymbol{\theta}_{n+1})\Delta\boldsymbol{\theta}_{n+1} \, , \tag{52}$$

$$\Delta\boldsymbol{\theta}_{n+1} = h\Delta\dot{\boldsymbol{\theta}}^{(\sigma)}_{n+1} + h^2\beta\Delta\mathbf{a}_{n+1} \, , \tag{53}$$

$$\Delta\dot{\boldsymbol{\theta}}^{(\sigma)}_{n+1} = \sigma\tfrac{\beta}{\gamma}\left(\Delta\dot{\boldsymbol{\theta}}_{n+1} - \Delta\mathbf{v}_{n+1}\right) \, , \tag{54}$$

$$\Delta\mathbf{v}_{n+1} = h\gamma\Delta\mathbf{a}_{n+1} \, , \tag{55}$$

$$(1-\alpha_m)\Delta\mathbf{a}_{n+1} = (1-\alpha_f)\Delta\dot{\mathbf{v}}_{n+1} \, , \tag{56}$$

$$\mathbf{T}(\boldsymbol{\theta}_{n+1})\Delta\dot{\boldsymbol{\theta}}_{n+1} = \Delta\mathbf{v}_{n+1} \, . \tag{57}$$

In the Newton–Raphson method, there are several choices for the unknowns, such as position, velocity or acceleration coordinates. In the following, we show how our σ-modified Lie group integrator can be implemented for the set of unknowns $(\Delta\boldsymbol{\theta}_{n+1}, \Delta\lambda_{n+1})$.

Rearranging Eqs. (53)–(55) yields

$$\sigma\tfrac{h\beta}{\gamma}\Delta\dot{\boldsymbol{\theta}}_{n+1} = \Delta\boldsymbol{\theta}_{n+1} - \tfrac{h\beta}{\gamma}(1-\sigma)\Delta\mathbf{v}_{n+1} \, . \tag{58}$$

Local coordinates and velocity increments are related by the tangent operator Eq. (57). Applying the tangent operator $\mathbf{T}(\boldsymbol{\theta}_{n+1})$ to Eq. (58) gives

$$\sigma\tfrac{h\beta}{\gamma}\Delta\mathbf{v}_{n+1} = \mathbf{T}(\boldsymbol{\theta}_{n+1})\Delta\boldsymbol{\theta}_{n+1} - \tfrac{h\beta}{\gamma}(1-\sigma)\mathbf{T}(\boldsymbol{\theta}_{n+1})\Delta\mathbf{v}_{n+1} \, . \tag{59}$$

[2] $\mathbb{R}^3 \times SO(3)$ or $SE(3)$ for example.

By rearranging Eq. (59) we obtain

$$\mathbf{T}(\boldsymbol{\theta}_{n+1})\Delta\boldsymbol{\theta}_{n+1} = \tfrac{h\beta}{\gamma}\mathbf{P}(\boldsymbol{\theta}_{n+1})\Delta\mathbf{v}_{n+1}\ , \tag{60}$$

with a matrix $\mathbf{P}$ that we define as

$$\mathbf{P}(\boldsymbol{\theta}) := \sigma\mathbf{I} + (1-\sigma)\mathbf{T}(\boldsymbol{\theta})\ . \tag{61}$$

Based on Eq. (60), we express $\Delta\mathbf{v}_{n+1}$ and $\Delta\dot{\mathbf{v}}_{n+1}$ as

$$\Delta\mathbf{v}_{n+1} = \gamma'\mathbf{P}^{-1}(\boldsymbol{\theta}_{n+1})\mathbf{T}(\boldsymbol{\theta}_{n+1})\Delta\boldsymbol{\theta}_{n+1}\ , \tag{62}$$

$$\Delta\dot{\mathbf{v}}_{n+1} = \beta'\mathbf{P}^{-1}(\boldsymbol{\theta}_{n+1})\mathbf{T}(\boldsymbol{\theta}_{n+1})\Delta\boldsymbol{\theta}_{n+1}\ , \tag{63}$$

where

$$\beta' = \frac{1-\alpha_m}{h^2\beta(1-\alpha_f)}\ , \quad \gamma' = \frac{\gamma}{h\beta}\ . \tag{64}$$

As can be seen from Eq. (61), matrix $\mathbf{P}$ involves the tangent operator $\mathbf{T}$. Hence, the analytical expression for $\mathbf{P}^{-1}$ depends on the specific structure of the tangent operator. For implementation, we approximate $\mathbf{P}^{-1}$ as[3]

$$\mathbf{P}^{-1}(\boldsymbol{\theta}) = \mathbf{I} + \frac{(1-\sigma)}{2}\left[\widehat{\boldsymbol{\theta}} - \frac{2-3(1-\sigma)}{6}\widehat{\boldsymbol{\theta}}^2\right] + \mathcal{O}(\|\boldsymbol{\theta}\|^3)\ , \quad (\|\boldsymbol{\theta}\| \to 0)\ . \tag{65}$$

For the Lie group $\mathbb{R}\times SO(3)$, we provide the matrix $\widehat{\boldsymbol{\theta}}$ in Appendix B. Inserting Eqs. (60)–(63) into Eqs. (46)–(47) and eliminating $\Delta\mathbf{q}_{n+1}$, $\Delta\mathbf{v}_{n+1}$, $\Delta\dot{\mathbf{v}}_{n+1}$ and $\Delta\lambda_{n+1}$ yields the linear system

$$\begin{bmatrix}\Delta\mathbf{r}\\ \Delta\boldsymbol{\Phi}\end{bmatrix} = \mathbf{S}_t\begin{bmatrix}\mathbf{T}\Delta\boldsymbol{\theta}\\ \Delta\lambda\end{bmatrix} \tag{66}$$

with the symmetric Jacobian

$$\mathbf{S}_t(\boldsymbol{\theta},\mathbf{q},\mathbf{v},\dot{\mathbf{v}},\lambda,t) = \begin{bmatrix}\mathbf{M}^*(\mathbf{q},\mathbf{v},t)\mathbf{P}^{-1}(\boldsymbol{\theta}) + \mathbf{K}_t(\mathbf{q},\mathbf{v},\dot{\mathbf{v}},\lambda,t) & \mathbf{B}^{\mathrm{T}}(\mathbf{q})\\ \mathbf{B}(\mathbf{q}) & \mathbf{0}\end{bmatrix}\ , \tag{67}$$

where

$$\mathbf{M}^*(\mathbf{q},\mathbf{v},t) := \beta'\mathbf{M}(\mathbf{q}) + \gamma'\mathbf{C}_t(\mathbf{q},\mathbf{v},t)\ . \tag{68}$$

It is known that the conditioning of the Jacobian of Newmark-type time integration methods is poor for very small time step sizes h. To achieve an optimal conditioning of the Jacobian Eq. (67), a suitable scaling strategy can be used, see e.g. [2,29].

As can be seen from Eq. (66), the tangent operator $\mathbf{T}$ can be moved from the Jacobian into the vector of unknowns. As a result, the local coordinates increment obtained from solving the linear system Eq. (66) must be multiplied by the inverse tangent operator $\mathbf{T}^{-1}$ to compute the local coordinates increment $\Delta\boldsymbol{\theta}$. Hence, for Newton iteration $j \to j+1$, it follows that

$$\boldsymbol{\theta}_{n+1}^{j+1} = \boldsymbol{\theta}_{n+1}^{j} + \mathbf{T}^{-1}(\boldsymbol{\theta}_{n+1}^{j})\Delta\boldsymbol{\theta}^{j+1}\ , \tag{69}$$

$$\mathbf{q}_{n+1}^{j+1} = \tau(\mathbf{q}_n, \boldsymbol{\theta}_{n+1}^{j+1})\ , \tag{70}$$

$$\mathbf{v}_{n+1}^{j+1} = \mathbf{v}_{n+1}^{j} + \gamma'\mathbf{P}^{-1}(\boldsymbol{\theta}_{n+1}^{j+1})\mathbf{T}(\boldsymbol{\theta}_{n+1}^{j+1})\mathbf{T}^{-1}(\boldsymbol{\theta}_{n+1}^{j})\Delta\boldsymbol{\theta}^{j+1}\ , \tag{71}$$

$$\dot{\mathbf{v}}_{n+1}^{j+1} = \dot{\mathbf{v}}_{n+1}^{j} + \beta'\mathbf{P}^{-1}(\boldsymbol{\theta}_{n+1}^{j+1})\mathbf{T}(\boldsymbol{\theta}_{n+1}^{j+1})\mathbf{T}^{-1}(\boldsymbol{\theta}_{n+1}^{j})\Delta\boldsymbol{\theta}^{j+1}\ , \tag{72}$$

$$\lambda_{n+1}^{j+1} = \lambda_{n+1}^{j} + \Delta\lambda^{j+1}\ . \tag{73}$$

Assuming that

$$\mathbf{T}(\boldsymbol{\theta}_{n+1}^{j+1})\mathbf{T}^{-1}(\boldsymbol{\theta}_{n+1}^{j}) \approx \mathbf{I}\ , \tag{74}$$

Eqs. (71)–(72) may be simplified as

$$\mathbf{v}_{n+1}^{j+1} = \mathbf{v}_{n+1}^{j} + \gamma'\mathbf{P}^{-1}(\boldsymbol{\theta}_{n+1}^{j+1})\Delta\boldsymbol{\theta}^{j+1}\ , \tag{75}$$

$$\dot{\mathbf{v}}_{n+1}^{j+1} = \dot{\mathbf{v}}_{n+1}^{j} + \beta'\mathbf{P}^{-1}(\boldsymbol{\theta}_{n+1}^{j+1})\Delta\boldsymbol{\theta}^{j+1}\ . \tag{76}$$

The simplification Eq. (74) reduces the complexity of the proposed approach, but could be critical. Our numerical experiments indicate that this simplification has a negligible impact on accuracy and does not affect the number of Newton iterations.

[3] If $(\Delta\mathbf{v}_{n+1}, \Delta\lambda_{n+1})$ are chosen as unknowns, $\mathbf{P}^{-1}$ does not have to be computed. Only $\mathbf{P}$ is involved in the Newton–Raphson method.

6.3. The case $\sigma = 1$

A particularly simple variant of our σ-modified Lie group generalized-α methods follows for $\sigma = 1$. In this case, the matrix $\mathbf{P}$ in Eq. (61) reduces to the identity matrix, and the dependence of the Jacobian in Eq. (67) as well as Eqs. (75)–(76) on the local coordinates is completely eliminated. It is worth noting that the tangent operator is not involved within the Jacobian in this case. As will become apparent in the next section and ultimately in Alg. 1, this yields a simple and computationally efficient implementation.

6.4. Updating absolute coordinates

In this section, we present an alternative approach to updating absolute coordinates with Eqs. (69)–(70). As we will show, the presented approach eliminates the need to evaluate the inverse tangent operator in Eq. (69). This is advantageous for $\sigma = 1$, because it results in a corrector step that is entirely free of both local coordinates and tangent operators.

We rewrite Eq. (70) as

$$\mathbf{q}_{n+1}^{j+1} = \boldsymbol{\tau}(\mathbf{q}_{n+1}^{j}, \Delta\mathbf{q}^{j+1}) \,, \tag{77}$$

where the increment $\Delta\mathbf{q}^{j+1}$ is obtained from the linearization of Eq. (70)

$$\Delta\mathbf{q}^{j+1} = \mathbf{T}(\boldsymbol{\theta}_{n+1}^{j+1})\Delta\boldsymbol{\theta}^{j+1} \,. \tag{78}$$

As shown in Eq. (66), the tangent operator $\mathbf{T}(\boldsymbol{\theta}_{n+1}^{j})$ can be shifted from the iteration matrix to the vector of unknowns. Thus, the increment obtained from Eq. (66) must be multiplied by $\mathbf{T}^{-1}(\boldsymbol{\theta}_{n+1}^{j})$ to obtain $\Delta\boldsymbol{\theta}^{j+1}$. Applying this in Eq. (78) and utilizing the assumption Eq. (74) once again, we derive the increment $\Delta\mathbf{q}^{j+1}$ as

$$\Delta\mathbf{q}^{j+1} = \mathbf{T}(\boldsymbol{\theta}_{n+1}^{j+1})\mathbf{T}^{-1}(\boldsymbol{\theta}_{n+1}^{j})\Delta\boldsymbol{\theta}^{j+1} = \Delta\boldsymbol{\theta}^{j+1} \,. \tag{79}$$

Inserting Eq. (79) into Eq. (77), we obtain the desired tangent operator free representation of Eq. (70):

$$\mathbf{q}_{n+1}^{j+1} = \boldsymbol{\tau}(\mathbf{q}_{n+1}^{j}, \Delta\boldsymbol{\theta}^{j+1}) \,. \tag{80}$$

6.5. Initializing the predictor step

The initialization of velocities $\mathbf{v}_{n+1}$, accelerations $\dot{\mathbf{v}}_{n+1}$, and algorithmic accelerations $\mathbf{a}_{n+1}$ in the predictor step of the proposed Newton–Raphson algorithms follows the standard procedure for generalized-α methods; see [2,12]. The nonlinear relationship between the time derivative of the local coordinates, $\dot{\boldsymbol{\theta}}_{n+1}$, and the velocities, $\mathbf{v}_{n+1}$, as described in Eq. (16), may necessitate a different initialization approach for $\boldsymbol{\theta}_{n+1}$ to ensure optimal performance. This section introduces the proposed initialization strategy.

For deriving the predictor equations for $\boldsymbol{\theta}_{n+1}$, we rewrite Eq. (16) as

$$\dot{\boldsymbol{\theta}}_{n+1} = \mathbf{T}^{-1}(\boldsymbol{\theta}_{n+1})\mathbf{v}_{n+1} \,. \tag{81}$$

Inserting Eq. (81) into Eq. (12) yields

$$\boldsymbol{\theta}_{n+1} = h\mathbf{v}_n + \sigma\frac{h\beta}{\gamma}(\mathbf{T}^{-1}(\boldsymbol{\theta}_{n+1})\mathbf{v}_{n+1} - \mathbf{v}_{n+1}) + h^2(\tfrac{1}{2} - \beta)\mathbf{a}_n + h^2\beta\mathbf{a}_{n+1} \,. \tag{82}$$

To resolve the nonlinearity in Eq. (82), we approximate $\mathbf{T}^{-1}(\boldsymbol{\theta})$ linearly as shown in Eq. (29). Inserting Eq. (29) into Eq. (82) and using the relation

$$\widehat{\boldsymbol{\theta}}_{n+1}\mathbf{v}_{n+1} = -\widehat{\mathbf{v}}_{n+1}\boldsymbol{\theta}_{n+1} \,, \tag{83}$$

yields after rearranging the proposed predictor

$$\boldsymbol{\theta}_{n+1} = \left(\mathbf{I} + \sigma\frac{h\beta}{\gamma}\widehat{\mathbf{v}}_{n+1}\right)^{-1}\left(h\mathbf{v}_n + h^2\left(\frac{1}{2} - \beta\right)\mathbf{a}_n + h^2\beta\mathbf{a}_{n+1}\right) \,. \tag{84}$$

For efficient computations, we approximate the inverse in Eq. (84) as

$$\left(\mathbf{I} + \sigma\frac{h\beta}{\gamma}\widehat{\mathbf{v}}_{n+1}\right)^{-1} \approx \left(\mathbf{I} - \sigma\frac{h\beta}{\gamma}\widehat{\mathbf{v}}_{n+1}\right) \,. \tag{85}$$

6.6. Newton–Raphson algorithm summarized for $\sigma = 1$

Algorithm Alg. 1 outlines the computation of a single step using the proposed index-3 Lie group generalized-α method for $\sigma = 1$. For $\sigma = \gamma/(3\beta)$, a algorithm is given in Appendix C and for the stabilized index-2 formulation in Appendix D.

To implement the σ-modified Lie group generalized-α method with $\sigma = 1$ in an existing multibody code that already includes geom1, only three modifications are required. First, line 6 from Alg. 1 must be added. Second, the tangent operator in the geom1 Jacobian must be removed. Finally, the Newton update for the absolute coordinates must be adapted as shown in line 15 of Alg. 1.

Algorithm 1 Numerical algorithm computing a Newton update for the proposed index-3 formulation for $\sigma = 1$.

```
function SolveTimeStepSigma1($\mathbf{q}_n$, $\mathbf{v}_n$, $\dot{\mathbf{v}}_n$, $\mathbf{a}_n$)
 1: $\dot{\mathbf{v}}_{n+1} := \mathbf{0}$
 2: $\lambda_{n+1} := \mathbf{0}$
 3: $\mathbf{a}_{n+1} := (\alpha_f \dot{\mathbf{v}}_n - \alpha_m \mathbf{a}_n)/(1-\alpha_m)$
 4: $\mathbf{v}_{n+1} := \mathbf{v}_n + h(1-\gamma)\mathbf{a}_n + h\gamma\mathbf{a}_{n+1}$
 5: $\boldsymbol{\theta}_{n+1} := h\mathbf{v}_n + h^2\left(\frac{1}{2}-\beta\right)\mathbf{a}_n + h^2\beta\mathbf{a}_{n+1}$
 6: $\boldsymbol{\theta}_{n+1} := \left(\mathbf{I} - \frac{h\beta}{\gamma}\widehat{\mathbf{v}}_{n+1}\right)\boldsymbol{\theta}_{n+1}$
 7: $\mathbf{q}_{n+1} := \boldsymbol{\tau}(\mathbf{q}_n, \boldsymbol{\theta}_{n+1})$
 8: for $j = 1$ to $j_{max}$ do
 9:     $\mathbf{res} = \begin{bmatrix} \mathbf{r}(\mathbf{q}_{n+1}, \mathbf{v}_{n+1}, \dot{\mathbf{v}}_{n+1}, \lambda_{n+1}, t_{n+1}) \\ \boldsymbol{\Phi}(\mathbf{q}_{n+1}) \end{bmatrix}$
10:     if $\|\mathbf{res}\| < tol$ then
11:         break
12:     else
13:         $\mathbf{S}_t := \mathbf{S}_t(\mathbf{q}_{n+1}, \mathbf{v}_{n+1}, \dot{\mathbf{v}}_{n+1}, \lambda_{n+1}, t_{n+1})$
14:         $\begin{bmatrix} \Delta_{\boldsymbol{\theta}} \\ \Delta_\lambda \end{bmatrix} := -\mathbf{S}_t^{-1}\,\mathbf{res}$
15:         $\mathbf{q}_{n+1} := \boldsymbol{\tau}(\mathbf{q}_{n+1}, \Delta_{\boldsymbol{\theta}})$
16:         $\mathbf{v}_{n+1} := \mathbf{v}_{n+1} + \gamma' \Delta_{\boldsymbol{\theta}}$
17:         $\dot{\mathbf{v}}_{n+1} := \dot{\mathbf{v}}_{n+1} + \beta' \Delta_{\boldsymbol{\theta}}$
18:         $\lambda_{n+1} := \lambda_{n+1} + \Delta_\lambda$
19:     end if
20: end for
21: $\mathbf{a}_{n+1} := \mathbf{a}_{n+1} + ((1-\alpha_f)/(1-\alpha_m))\dot{\mathbf{v}}_{n+1}$
22: return $\mathbf{q}_{n+1}$, $\mathbf{v}_{n+1}$, $\dot{\mathbf{v}}_{n+1}$, $\mathbf{a}_{n+1}$, $\lambda_{n+1}$,
end function
```

7. Numerical examples

In this section, we compare the proposed σ-modified Lie group generalized-α methods with both the geom1 method and conventional non-Lie group formulations based on Euler parameters and/or Cardan (Tait-Bryan) angles. In [13,30], the accuracy and computational performance of the geom1 method were evaluated against the conventional generalized-α method using several rigid multibody systems. To evaluate our σ-modified Lie group generalized-α methods, some of the rigid body examples considered here are drawn from [13,30]. The following abbreviations are used:

- Gα (CTB): Cardan/Tait-Bryan angles formulation (index-3), cf. [13]
- Gα (EP): DAE-based Euler parameter formulation (index-3), cf. [13]
- geom1: Lie group generalized-α (index-2, index-3, stabilized index-2), see [14,20]
- geom1-σ_1: Proposed approach with $\sigma = 1$ (index-2, index-3, stabilized index-2)
- geom1-$\sigma_{\frac{\gamma}{3\beta}}$: Proposed approach with $\sigma = \gamma/(3\beta)$ (index-2, index-3, stabilized index-2)

All numerical examples have been implemented in the open source multibody simulation code Exudyn [31] using version 1.10.6 and Python 3.13 64 bit. Note that the conventional geom1 method (index-2, index-3), as well as the proposed extension with $\sigma = 1$ (index-2, index-3), have been implemented in C++ with a focus on efficiency. The proposed σ-modified methods with $\sigma = \gamma/(3\beta)$ (index-2, index-3, stabilized index-2) have been implemented in Python and integrated into Exudyn via the SetUserFunctionNewton() interface. In addition, the proposed stabilized index-2 method for $\sigma = 1$ has also been implemented through SetUserFunctionNewton(). All simulations have been performed under Windows 10 on an Intel Core i7-6600U 2.60 GHz processor. Unless explicitly stated, the absolute and relative tolerances a_{tol} and r_{tol} in the Newton method were selected as $a_{tol} = 1.0 \times 10^{-10}$ and $r_{tol} = 1.0 \times 10^{-8}$ and the spectral radius as $\rho_\infty = 0.9$. The label "Simplified Jacobian" refers to cases where the matrix $\mathbf{P}^{-1}$ (used in the Jacobian for the $\sigma \neq 1$ methods) and the tangent operator $\mathbf{T}$ (used in the geom1 methods) are omitted. Conversely, "Full Jacobian" indicates that both $\mathbf{P}^{-1}$ and $\mathbf{T}$ are fully included in the Jacobian matrices. For the C++ implementations, Exudyn provides a modified Newton method, which will also be used for performance evaluation. In this method, the iteration matrix is computed once at the beginning of the simulation and updated only when the contractivity becomes larger than 0.5. Specifically, an update and factorization is computed if either a predefined maximum number of Newton iterations is exceeded or if the error cannot be reduced below a prescribed tolerance in a given iteration; see [31,32]. For easier comparison with the conventional Euler parameter formulation, we have evaluated the error in the rotation parameters in terms of Euler parameters.

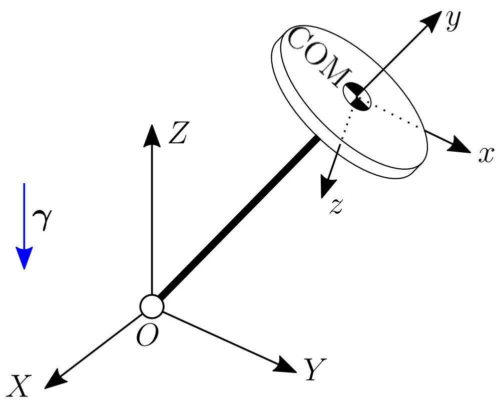


Fig. 1. Benchmark problem Heavy top.

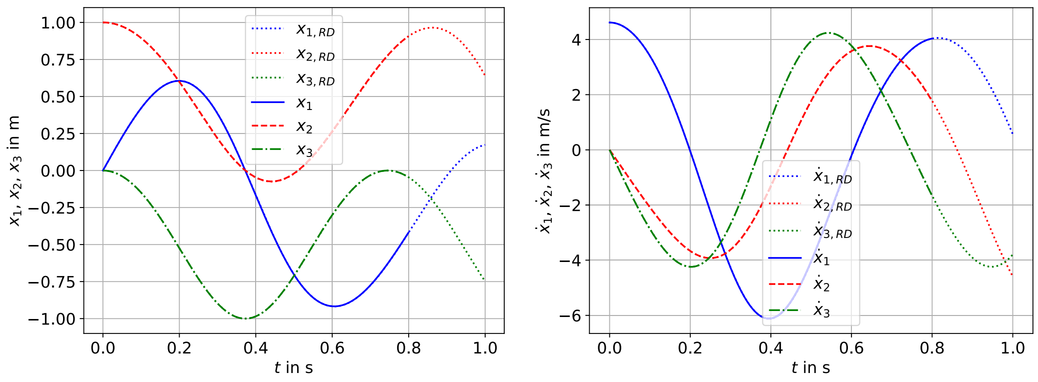


Fig. 2. Time histories of the heavy top's center of mass (COM) position (left plot) and translational velocity coordinates (right plot) expressed in the global frame. Reference solutions were computed over a simulation time of $t = 1$ s with a time step size of $h = 2.5\times10^{-5}$ s using RecurDyn (RD). The heavy top's center of mass position and translational velocity coordinates obtained with Euler parameters and generalized-α are computed for comparison purpose over $t = 0.8$ s.

7.1. Heavy top with kinematic constraints

The heavy top, illustrated in Fig. 1, is a classical benchmark used to evaluate Lie group time integration methods (see, e.g., [12,33]). The system consists of a single rigid body attached to a fixed point O via a spherical joint and subject to gravity. The motion of the heavy top could, in principle, be described without kinematic constraints purely on SO(3) if the equilibrium of momentum were formulated with respect to the fixed point, see, e.g., [33]. To evaluate our algorithms for constrained systems, we consider the translation of the center of mass $\mathbf{x} \in \mathbb{R}^3$ and the rotation of the body $\boldsymbol{\psi} \in \mathbb{R}^3$ as independent variables and model the heavy top using the $\mathbb{R}^3 \times SO(3)$ formulation, cf. [22]. Model parameters and initial conditions can be found, for example, in [12,13,33].

In Fig. 2, the time histories of the heavy top's center of mass position coordinates (left plot) and translational velocity coordinates (right plot) computed with the proposed index-3 Lie group generalized-α method are compared with reference solutions computed with the multibody simulation software RecurDyn. As can be seen in Fig. 2, the time histories obtained with the proposed Lie group generalized-α method visually match the reference solutions. The absolute deviation of the heavy top's position and velocity coordinates at $t = 1$ s is approximately 1.0×10^{-5} m respectively 1.0×10^{-5} m/s. Similar results have been obtained for the proposed index-2 integrator.

In Fig. 3, the convergence in the norm of the absolute coordinates (Fig. 3(a)) and velocities (Fig. 3(b)) are illustrated for different time integration methods. For the index-3 integrators, the spectral radius was selected as $\rho_\infty = 0.65$ and for the stabilized index-2 methods as $\rho_\infty = 1.0$. In Fig. 4, the convergence in the norm of the Lagrange multipliers error is illustrated. As shown in Figs. 3 and 4, the σ-modified generalized-α methods exhibit the expected higher accuracy compared to geom1 and outperform both conventional formulations. For clarity, the figures omit the results for the index-2 methods. Nevertheless, the proposed index-2 σ-modified methods exhibit a similar accuracy improvement as the corresponding index-3 methods.

With full Jacobians, the average number of Newton iterations per time step is 2 for all Lie group methods, 3.8 for the conventional Euler parameter method, and 2.1 for the Cardan–Tait–Bryan angle method. While the σ-modified generalized-α methods with $\sigma = 1$ do not require any tangent-operator–like matrix in the Jacobian, such matrices can also be neglected in geom1 and in the σ-modified

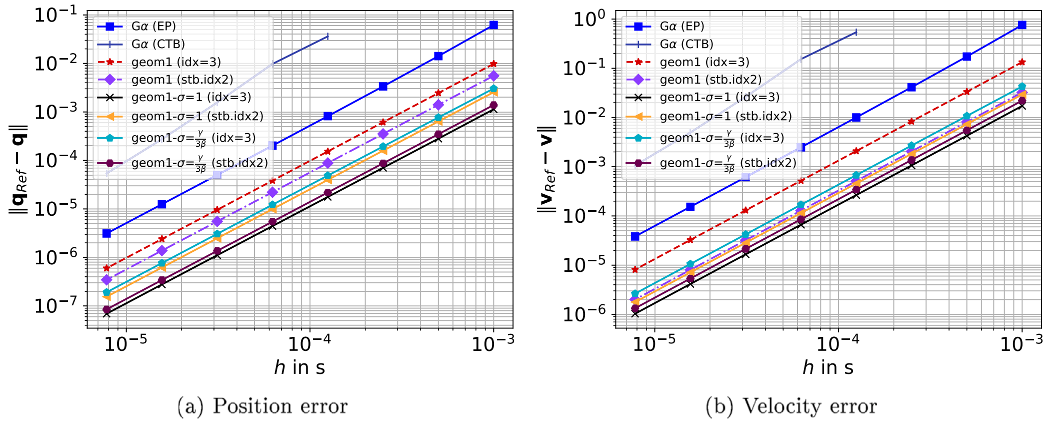


Fig. 3. Convergence in the norm of the maximum error of the heavy top's absolute coordinates and velocities for different time integration methods.

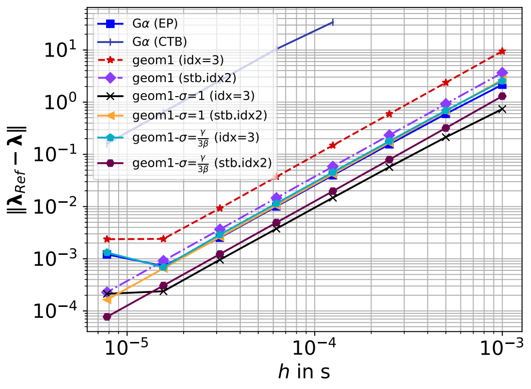


Fig. 4. Convergence in the norm of the maximum error of the Lagrange multipliers for different time integration methods in the heavy top example.

generalized-α methods for $\sigma \neq 1$, which simplifies their implementation. With simplified Jacobians, the average number of Newton iterations per time step is 3.1 for geom1 (index-3), 2.9 for geom1 (stabilized index-2), 2.5 for the proposed σ-modified method (index-3, $\sigma = \gamma/(3\beta)$), and 3.4 for its stabilized index-2 variant. In case modified Newton is used, the mean number of Newton iterations per time step is equal to 7.9 for both the proposed Lie group integrators with $\sigma = 1$ and for geom1 (index-2, index-3), and equal to 8.4 for the conventional Euler parameter method and 7.3 for the Cardan-Tait/Bryan method. It should be noted that the comparatively low average iteration number of the Cardan–Tait–Bryan formulation arises because it fails to simulate the heavy top at the three largest time steps, but succeeds at smaller step sizes where fewer Newton iterations are required.

7.2. High-speed rotor with flexible supports

The high-speed rotor with flexible supports depicted in Fig. 5 was used in [13] to evaluate the performance of Lie group time integration methods and it turned out that the conventional Euler parameter formulation achieves a higher accuracy as geom1, which motivates this example. Gravity is not considered but the rotor is subjected to a constant torque:

$$\mathbf{t} = \begin{bmatrix} 0 & 0 & -(\frac{L}{2} - L_L)mg \end{bmatrix}^{\mathrm{T}}. \tag{86}$$

The model parameters are given in Table 1.

In Fig. 6, the trajectory of point $\mathbf{p}_1 = [-L_L, 0.0, 0.0]$ in y-z-plane (left plot) and the time history of its coordinates (right plot) are shown. Fig. 6 was obtained using the proposed index-3 Lie group integrator with a time step size of $h = 1.0 \times 10^{-5}$ s. As can be seen in the left plot of Fig. 6, the rotor performs a precession motion, whereby the precession period is about 2.5 s (cf. Fig. 6 (right plot)), which corresponds to a precession frequency of 0.4 Hz.

In Fig. 7, the convergence in the norm of the right bearing position (Fig. 7(a)) and velocities (Fig. 7(b)) are illustrated for different time integration methods. Convergence is investigated at $t = 1$ s. As shown in Fig. 7(a), the σ-modified Lie group generalized-α methods achieve an accuracy approximately one order of magnitude higher than that of the geom1 methods. A similar improvement

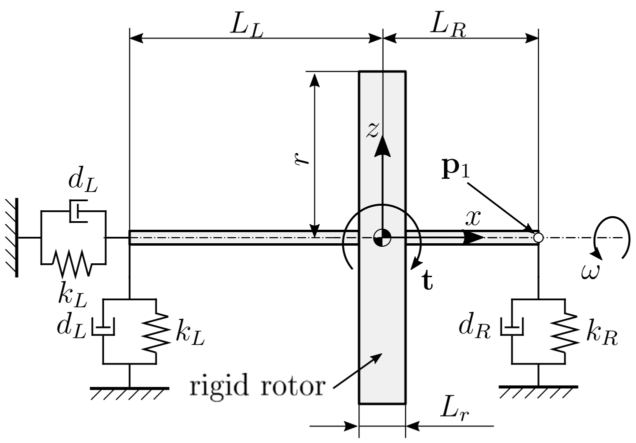


Fig. 5. Schematic representation of high-speed rotor with flexible supports.

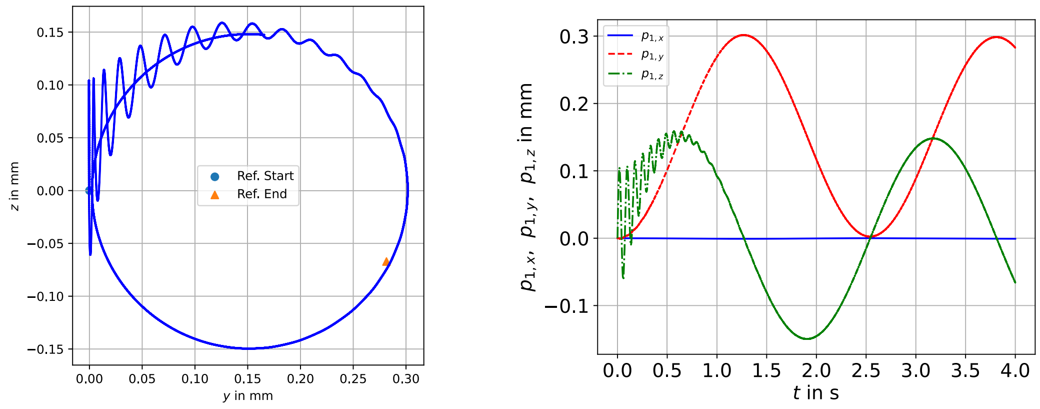
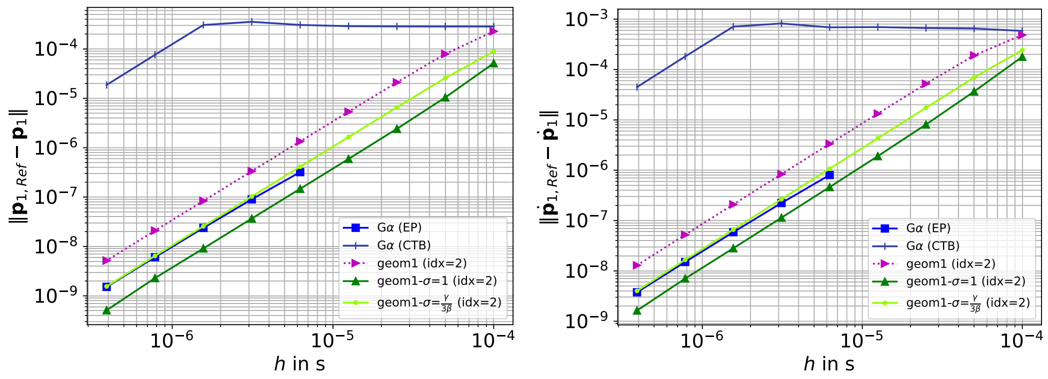


Fig. 6. *Left*: Trajectory of point $\mathbf{p}_1$ on the rotor located at the position of the left support. *Right*: Time history of the components of angular velocity vector represented in the local frame. *Left plot* Tilting motion of the rotor, *right plot* position coordinates of point $\mathbf{p}_1$.

Table 1
Model parameters of 'high-speed rotor with flexible supports'.

Model parameter	Value	Notes
m in kg	1.223	mass of the rotor
r in m	0.05	radius for disk mass distribution
L_r in m	0.02	length of rotor disk
J_{xx} in kg m^2	0.001541	polar moment of inertia
J_{yy} in kg m^2	0.000812	moment of inertia for y axes
J_{zz} in kg m^2	0.000812	moment of inertia for z axes
L_L in m	0.11	length of rotor
L_R in m	0.09	length of rotor
n in rpm	200 000	rotational speed
ω in rad/s	20 944	$\omega = 2\pi n/60$
k_L in N/m	4000	applied in x-y-z-direction
k_R in N/m	4000	applied in y-z-direction
D	0.0001	dimensionless damping
d_L in kg/s	5.165093	$d_L = 2D\omega m$ applied in x-y-z-direction
d_R in kg/s	5.165093	$d_R = 2D\omega m$ applied in y-z-direction
g in m/s^2	9.81	gravitational acceleration



(a) Position error (b) Velocity error

Fig. 7. Convergence in the norm of the maximum error of the right bearing position and velocity.

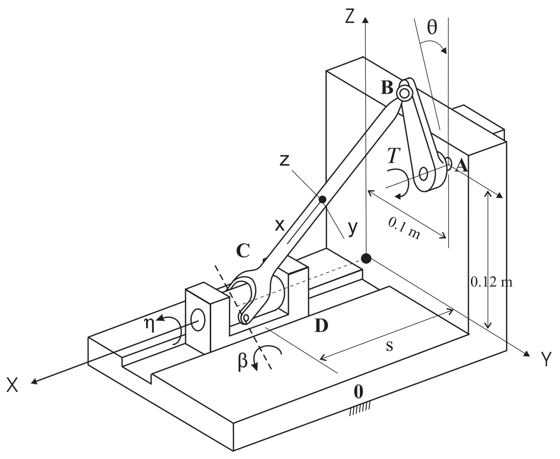


Fig. 8. Spatial rigid slider-crank mechanism [34,35].

is observed for the velocity of the left bearing point (see Fig. 7(b)). As shown in Fig. 7, the σ-modified Lie group generalized-α method achieves higher accuracy than the conventional Euler parameter formulation and substantially higher accuracy than the Cardan–Tait/Bryan formulation. It is worth noting that all Lie group integrators successfully simulate the high-speed rotor even at the largest time step sizes considered, whereas the conventional Euler parameter formulation fails, see Fig. 7. While the Cardan–Tait–Bryan angle formulation remains stable at the largest time step sizes, its accuracy is significantly lower than that of the Lie group integrators.

In case of full and simplified Jacobians, the mean number of Newton iterations per time step is equal to 1.3 for the index-2 σ-modified generalized-α methods, equal to 1.8 for geom1 with index-2, and equal to 2.56 for the conventional Euler parameter and Cardan-Tait/Bryan angle method. When modified Newton is used, the mean number of Newton iterations per time step is 2.1 for the proposed σ-modified Lie group integrators (index-2 and index-3), 2.3 for geom1 (index-2 and index-3), 3.3 for the conventional Cardan–Tait/Bryan method, and 8.3 for the conventional Euler parameter method. We want to note, that all Lie group integration methods required on average only one Jacobian evaluation for the entire simulation with modified Newton, whereas the conventional Euler parameter method required 119823 Jacobian updates.

7.3. Spatial rigid slider-crank mechanism

The spatial slider-crank mechanism [34], schematically illustrated in Fig. 8, was also studied in [13] to evaluate the performance of Lie group time integration methods. It was shown that the conventional Euler parameter formulation achieves, on average, higher accuracy than the geom1 method, which motivates the use of this example.

The spatial slider-crank mechanism consists of four rigid bodies: slider, crank, connecting rod and ground. The crank (AB) has a length of 0.08 m and is connected to ground by a revolute joint at point A. The crank is driven from initial crank angle $\theta = 0$ rad with an initial angular velocity of 6 rad/s. The connecting rod has a length of 0.3 m and is connected to the crank by a spherical joint at point B and to the slider by a universal joint at point C. The slider is connected to ground by a prismatic joint at point

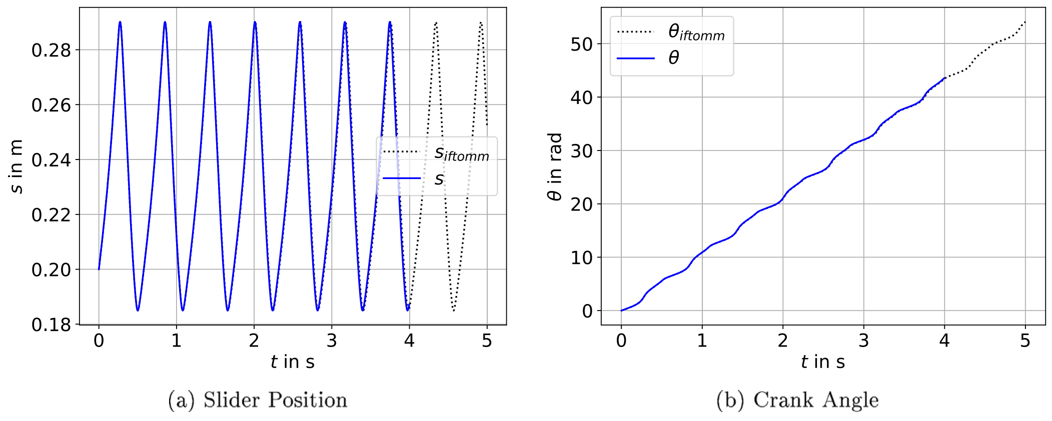


Fig. 9. Time histories of the slider crank's slider position and crank angle. The reference solution ($s_{Masarati}$) was taken from the library of computational benchmark problems IFToMM [34].

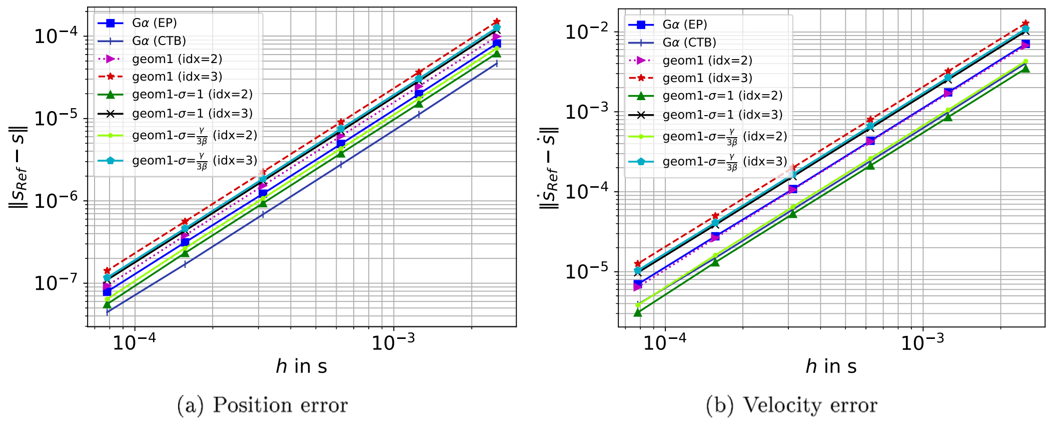


Fig. 10. Convergence in the norm of the maximum error of the slider crank's slider position and slider velocity for different time integration methods.

D with sliding displacement s. All links are subjected to gravity of magnitude 9.81 m/s^2 in the negative z-direction. Studying the dynamic response of the slider-crank mechanism under the gravitational force is the main objective of this benchmark problem. Model parameters and initial conditions are given in [13,34]. The relative tolerance r_{tol} in the Newton method was selected as $r_{tol} = 1.0 \times 10^{-9}$ and the spectral radius as $\rho_\infty = 0.875$.

In Fig. 9, the time histories of the slider-crank's slider position coordinate s and crank angle θ are compared with reference solutions provided by IFToMM [34]. Fig. 9 was obtained using the proposed index-3 Lie group generalized-α method with $\sigma = 1$ and time step size $h = 1.0 \times 10^{-5}$ s. As can be seen in Fig. 9, the time histories obtained with generalized-α and Euler parameters visually match the reference solution. The absolute deviation of the slider crank's slider position s at $t = 5$ s is approximately 8.6×10^{-3} m. The difference between our solution and the solution provided by IFToMM is due to the fact that different time integration methods were used, see [34].

In Fig. 10, the convergence in the norm of the slider crank's slider position and slider velocity error, evaluated at $t = 2$ s, is illustrated for different time integration methods. As can be seen in Fig. 10, the proposed σ-modified Lie group methods yield higher accuracy as the geom1 methods. The proposed index-2 integrators exhibit in this example higher accuracy as the conventional Euler parameter method. In contrast to the previous examples, the conventional Cardan-Tait/Bryan angles formulation turns out to be the most accurate one in this example.

When full and simplified Jacobians are used, the mean number of Newton iterations per time step is 2.5 for the index-2 σ-modified generalized-α methods and geom1, 2.0 for the index-3 σ-modified generalized-α methods and geom1, 3.0 for the conventional Euler parameter method, and 2.0 for the Cardan–Tait/Bryan angle method. When modified Newton is used, the mean number of Newton iterations per time step is 7.2 for both the proposed σ-modified Lie group integrators and geom1 (index-2 and index-3), 7.8 for the conventional Euler parameter method, and 7.3 for the conventional Cardan–Tait/Bryan method. Note, all Lie group integration methods required on average 170 Jacobian evaluations for the entire simulation with modified Newton, whereas the conventional Euler parameter method required 1000 Jacobian updates. We want to point out that for the σ-modified integrator with $\sigma = 1$, the

Jacobian structure is identical to the one obtained by neglecting the matrix $\mathbf{P}^{-1}$ in the σ-modified integrators with $\sigma \neq 1$ and the tangent operator $\mathbf{T}$ in the Jacobian of geom1. This shows the increased simplicity of the proposed σ-modified integrators with $\sigma = 1$ compared to geom1.

8. Conclusion

This paper introduces novel σ-modified Lie group generalized-α methods for the simulation of constrained multibody systems. We demonstrate second-order accuracy for differential–algebraic equations both analytically and through numerical examples involving several constrained rigid and flexible multibody systems. A detailed analysis of the local truncation error shows that the σ-modification does not affect the order of the Lie group integrator but enables the elimination of a Lie group-specific part of the leading error term. The proposed methods inherit the favorable zero-stability properties of classical generalized-α Lie group integrators, both for unconstrained systems and for constrained systems in index-3, index-2, or stabilized index-2 formulations.

The σ-modified Lie group generalized-α methods are benchmarked against a conventional Euler parameter and Cardan-Tait/Bryan angles formulation as well as the geom1 Lie group generalized-α method [20], which is the most frequently used generalized-α Lie group integrator in flexible multibody dynamics. Based on the numerical results, we conclude that for constrained rigid body systems with large rotations, the proposed σ-modified Lie group generalized-α methods deliver higher accuracy than the original geom1 method and comparable or superior accuracy to the conventional Euler parameter formulation. In most of the investigated examples, the conventional Cardan–Tait/Bryan angle formulation was outperformed by every other method considered. For flexible multibody systems modeled using the floating frame of reference formulation, we observed that the accuracy of the σ-modified methods is comparable to that of geom1 and the conventional Euler parameter formulation.

Comparing the computational effort for one step for the special case $\sigma = 1$, the σ-modified methods outperform geom1. This is because the corrector stage does not require handling of local coordinates in this case — neither in the Jacobian nor in the Newton updates. The performance advantage becomes also apparent when simplified Jacobians are used in geom1, as the Jacobian of the proposed method with $\sigma = 1$, by design, does not involve the tangent operator. The absence of the tangent operator preserves the sparsity structure of the system matrices and avoids potential stability issues that may arise in geom1 when the tangent operator is excluded. Moreover, depending on the chosen coordinate map and the Lie group used for modeling, evaluating the tangent operator can be computationally expensive and prone to numerical issues, cf. [17]. Therefore, tangent-operator-free Lie group integrators – such as the method with $\sigma = 1$ – offer clear advantages in terms of both efficiency and numerical robustness. The improvements in computational efficiency and accuracy are particularly beneficial for real-time applications, large-scale simulations, parametric studies, and optimization tasks. In conclusion, the proposed σ-modified Lie group time integrators show strong potential for accurate, reliable and high-performance multibody simulations.

CRediT authorship contribution statement

Stefan Holzinger: Conceptualization, Formal analysis, Investigation, Methodology, Project administration, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Martin Arnold:** Conceptualization, Formal analysis, Investigation, Methodology, Writing – original draft, Writing – review & editing. **Johannes Gerstmayr:** Conceptualization, Methodology, Software, Writing – original draft, Writing – review & editing.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the author(s) used GPT-5 in order to improve the readability and language of the manuscript. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the published article.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. LGT maps for rigid and FFRF bodies

For a multibody system consisting of N bodies (rigid, flexible), the LGT map Eq. (44) collects N LGT maps for N bodies. For the ith rigid body in the multibody system, the LGT map reads

$$\begin{aligned} \boldsymbol{\tau}_{\mathrm{RB}}^{(i)} &: \mathbb{R}^{3+l} \times \mathbb{R}^{6} \to \mathbb{R}^{3+l} , \\ (\mathbf{q}^{(i)}(t_n), \boldsymbol{\theta}^{(i)}(t)) &\mapsto \mathbf{q}^{(i)}(t) = \boldsymbol{\tau}_{\mathrm{RB}}^{(i)}(\mathbf{q}^{(i)}(t_n), \boldsymbol{\theta}^{(i)}(t)) , \end{aligned} \tag{A.1}$$

where the variable l in Eq. (A.1) denotes the number of rotation parameters used to globally parameterize SO(3). For the ith flexible body modeled with the floating frame of reference formulation [36,37], the LGT map reads

$$\begin{aligned} \boldsymbol{\tau}_{\mathrm{FFRF}}^{(i)} &: \mathbb{R}^{3+l+n_f} \times \mathbb{R}^{6+n_f} \to \mathbb{R}^{3+l+n_f} , \\ (\mathbf{q}^{(i)}(t_n), \boldsymbol{\theta}^{(i)}(t)) &\mapsto \mathbf{q}^{(i)}(t) = \boldsymbol{\tau}_{\mathrm{FFRF}}^{(i)}(\mathbf{q}^{(i)}(t_n), \boldsymbol{\theta}^{(i)}(t)) , \end{aligned} \tag{A.2}$$

where the variable n_f in Eq. (A.2) denotes the number of flexible (modal) coordinates used in the floating frame of reference formulation. The explicit form of the LGT maps Eqs. (A.1)–(A.2) depend on the Lie group chosen to describe the configuration space of a single body and the rotation parameters used to parameterize SO(3). In this paper, we use the (Cartesian) rotation vector $\boldsymbol{\psi} \in \mathbb{R}^3$ as absolute rotational coordinates. The rotation vector is defined as the vector

$$\boldsymbol{\psi} = \varphi \, \mathbf{n} \, , \tag{A.3}$$

which has the direction of the rotation axis $\mathbf{n} = \boldsymbol{\psi}/\|\boldsymbol{\psi}\|$ and a length equal to the rotation angle $\varphi = \|\boldsymbol{\psi}\|$, see, e.g., [9]. In our case, the absolute coordinates $\mathbf{q}^{(i)} \in \mathbb{R}^6$ for the ith rigid body are

$$\mathbf{q}^{(i)}_{\mathrm{RB}} = \left[\left(\mathbf{x}^{(i)}\right)^{\mathrm{T}} \quad \left(\boldsymbol{\psi}^{(i)}\right)^{\mathrm{T}} \right]^{\mathrm{T}} , \tag{A.4}$$

where $\mathbf{x}^{(i)} \in \mathbb{R}^3$ denotes the position of a rigid body w.r.t. a global inertial frame of reference. For the ith modally reduced FFRF-body, we have

$$\mathbf{q}^{(i)}_{\mathrm{FFRF}} = \left[\left(\mathbf{x}^{(i)}\right)^{\mathrm{T}} \quad \left(\boldsymbol{\psi}^{(i)}\right)^{\mathrm{T}} \quad \left(\boldsymbol{\zeta}^{(i)}\right)^{\mathrm{T}} \right]^{\mathrm{T}} , \tag{A.5}$$

where $\boldsymbol{\zeta}^{(i)}$ are modal coordinates.

For implementation purposes, it is convenient to separate the LGT maps Eqs. (A.1)–(A.2) into a LGT map for rigid body translations $\boldsymbol{\tau}_{\mathrm{t}}$

$$\begin{aligned} \boldsymbol{\tau}_{\mathrm{t}} \, : \mathbb{R}^3 \times \mathbb{R}^3 &\to \mathbb{R}^3, \\ (\mathbf{x}(t_n),\, \boldsymbol{\theta}_{\mathrm{t}}(t)) &\mapsto \mathbf{x}(t) = \mathbf{x}(t_n) + \boldsymbol{\theta}_{\mathrm{t}}(t) \end{aligned} , \tag{A.6}$$

a LGT map for rigid body rotations $\boldsymbol{\tau}_{\mathrm{r}}$

$$\begin{aligned} \boldsymbol{\tau}_{\mathrm{r}} \, : \mathbb{R}^3 \times \mathbb{R}^3 &\to \mathbb{R}^3, \\ (\boldsymbol{\psi}(t_n),\, \boldsymbol{\theta}_{\mathrm{r}}(t)) &\mapsto \boldsymbol{\psi}(t) = \boldsymbol{\psi}(t_n) \diamond \boldsymbol{\theta}_{\mathrm{r}}(t) \, . \end{aligned} \tag{A.7}$$

and in the case of a FFRF-body into a LGT map for the flexible coordinates

$$\begin{aligned} \boldsymbol{\tau}_{\mathrm{f}} \, : \mathbb{R}^{n_f} \times \mathbb{R}^{n_f} &\to \mathbb{R}^{n_f}, \\ (\boldsymbol{\zeta}(t_n),\, \boldsymbol{\theta}_{\mathrm{f}}(t)) &\mapsto \boldsymbol{\zeta}(t) = \boldsymbol{\zeta}(t_n) + \boldsymbol{\theta}_{\mathrm{f}}(t) \, . \end{aligned} \tag{A.8}$$

The symbol $\diamond$ in Eq. (A.7) denotes the composition operation for rotation vectors, which is given explicitly for example in Eq. (26) in [38].

Appendix B. Hat-operator matrices for $\mathbb{R} \times$ SO(3)

For implementation purposes, it is convenient to decompose $\boldsymbol{\theta}$ for each body in the multibody system into components associated with rigid body translations $\boldsymbol{\theta}_{\mathrm{t}} \in \mathbb{R}^3$, rotations $\boldsymbol{\theta}_{\mathrm{r}} \in \mathbb{R}^3$, and a flexible part $\boldsymbol{\theta}_{\mathrm{f}} \in \mathbb{R}^{n_f}$ in the case of a FFRF-body

$$\boldsymbol{\theta}_{\mathrm{RB}} = \left[{\boldsymbol{\theta}_{\mathrm{t}}}^{\mathrm{T}} \quad {\boldsymbol{\theta}_{\mathrm{r}}}^{\mathrm{T}} \right]^{\mathrm{T}}, \qquad \boldsymbol{\theta}_{\mathrm{FFRF}} = \left[{\boldsymbol{\theta}_{\mathrm{t}}}^{\mathrm{T}} \quad {\boldsymbol{\theta}_{\mathrm{r}}}^{\mathrm{T}} \quad {\boldsymbol{\theta}_{\mathrm{f}}}^{\mathrm{T}} \right]^{\mathrm{T}} . \tag{B.1}$$

For the Lie group $G = \mathbb{R}^3 \times SO(3)$ and a single rigid body, matrix $\widehat{\boldsymbol{\theta}}$ reads

$$\widehat{\boldsymbol{\theta}}_{\mathrm{RB}}(\boldsymbol{\theta}_{\mathrm{RB}}) = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \widetilde{\boldsymbol{\theta}}_{\mathrm{r}} \end{bmatrix} , \tag{B.2}$$

and for a single FFRF-body we have

$$\widehat{\boldsymbol{\theta}}_{\mathrm{FFRF}}(\boldsymbol{\theta}_{\mathrm{FFRF}}) = \begin{bmatrix} \widehat{\boldsymbol{\theta}}_{\mathrm{RB}}(\boldsymbol{\theta}_{\mathrm{RB}}) & \mathbf{0}_{6\times n_f} \\ \mathbf{0}_{n_f\times6} & \mathbf{0}_{n_f\times n_f} \end{bmatrix} , \tag{B.3}$$

where n_f is the number of flexible (modal) coordinates used.

Appendix C. Newton–Raphson algorithm summarized for $\sigma = \gamma/(3\beta)$

Algorithm Alg. 2 outlines the computation of a single step using the proposed index-3 Lie group generalized-α method for $\sigma = \gamma/(3\beta)$.

Appendix D. Newton–Raphson algorithm for stabilized-index-2

The derivation and implementation of the proposed index-3 σ-modified Lie group generalized-α methods were detailed in Section 6. While the initialization procedure introduced there also applies to the stabilized index-2 integrator, the corrector step in the stabilized index-2 scheme must be extended to account for the auxiliary variables $\boldsymbol{\eta}$. This section summarizes the key formulas necessary for implementing the stabilized index-2 σ-modified Lie group generalized-α method.

Algorithm 2 Numerical algorithm computing a Newton update for the proposed index-3 formulation for $\sigma = \gamma/(3\beta)$.

```
function SolveTimeStepSigmaOpt($\mathbf{q}_n$, $\mathbf{v}_n$, $\dot{\mathbf{v}}_n$, $\mathbf{a}_n$)
1: $\dot{\mathbf{v}}_{n+1} := \mathbf{0}$
2: $\lambda_{n+1} := \mathbf{0}$
3: $\mathbf{a}_{n+1} := (\alpha_f \dot{\mathbf{v}}_n - \alpha_m \mathbf{a}_n)/(1-\alpha_m)$
4: $\mathbf{v}_{n+1} := \mathbf{v}_n + h(1-\gamma)\mathbf{a}_n + h\gamma \mathbf{a}_{n+1}$
5: $\boldsymbol{\theta}_{n+1} := h\mathbf{v}_n + h^2\left(\frac{1}{2}-\beta\right)\mathbf{a}_n + h^2\beta\mathbf{a}_{n+1}$
6: $\boldsymbol{\theta}_{n+1} := \left(\mathbf{I} - \frac{h}{3}\widehat{\mathbf{v}}_{n+1}\right)\boldsymbol{\theta}_{n+1}$
7: $\mathbf{q}_{n+1} := \boldsymbol{\tau}(\mathbf{q}_n, \boldsymbol{\theta}_{n+1})$
8: for $j = 1$ to $j_{max}$ do
9:     $\mathbf{res} = \begin{bmatrix} \mathbf{r}(\mathbf{q}_{n+1}, \mathbf{v}_{n+1}, \dot{\mathbf{v}}_{n+1}, \lambda_{n+1}, t_{n+1}) \\ \boldsymbol{\Phi}(\mathbf{q}_{n+1}) \end{bmatrix}$
10:    if $\|\mathbf{res}\| < tol$ then
11:        break
12:    else
13:        $\mathbf{S}_t := \mathbf{S}_t(\boldsymbol{\theta}_{n+1}, \mathbf{q}_{n+1}, \mathbf{v}_{n+1}, \dot{\mathbf{v}}_{n+1}, \lambda_{n+1}, t_{n+1})$
14:        $\begin{bmatrix} \Delta_{\boldsymbol{\theta}} \\ \Delta_\lambda \end{bmatrix} := -\mathbf{S}_t^{-1}\,\mathbf{res}$
15:        $\boldsymbol{\theta}_{n+1} := \boldsymbol{\theta}_{n+1} + \mathbf{T}^{-1}(\boldsymbol{\theta}_{n+1})\Delta_{\boldsymbol{\theta}}$
16:        $\mathbf{q}_{n+1} := \tau(\mathbf{q}_n, \boldsymbol{\theta}_{n+1})$
17:        $\mathbf{v}_{n+1} := \mathbf{v}_{n+1} + \gamma'\mathbf{P}^{-1}(\boldsymbol{\theta}_{n+1})\Delta_{\boldsymbol{\theta}}$
18:        $\dot{\mathbf{v}}_{n+1} := \dot{\mathbf{v}}_{n+1} + \beta'\mathbf{P}^{-1}(\boldsymbol{\theta}_{n+1})\Delta_{\boldsymbol{\theta}}$
19:        $\lambda_{n+1} := \lambda_{n+1} + \Delta_\lambda$
20:    end if
21: end for
22: $\mathbf{a}_{n+1} := \mathbf{a}_{n+1} + ((1-\alpha_f)/(1-\alpha_m))\dot{\mathbf{v}}_{n+1}$
23: return $\mathbf{q}_{n+1}$, $\mathbf{v}_{n+1}$, $\dot{\mathbf{v}}_{n+1}$, $\mathbf{a}_{n+1}$, $\lambda_{n+1}$,
end function
```

For the stabilized index-2 formulation, the linear system to be solved reads

$$\begin{bmatrix} \Delta\mathbf{r} \\ \Delta\boldsymbol{\Phi} \\ \Delta\dot{\boldsymbol{\Phi}} \end{bmatrix} = \mathbf{S}_t \begin{bmatrix} \mathbf{T}\Delta\boldsymbol{\theta} \\ \Delta\lambda \\ \Delta\boldsymbol{\eta} \end{bmatrix} . \tag{D.1}$$

The Jacobian is given by

$$\mathbf{S}_t(\boldsymbol{\theta}, \mathbf{q}, \mathbf{v}, \dot{\mathbf{v}}, \lambda, t) = \begin{bmatrix} \mathbf{M}^*\mathbf{P}^{-1} + \mathbf{K}_t & \mathbf{B}^{\mathrm{T}} & \frac{\sigma}{\gamma'}\mathbf{M}^*\mathbf{P}^{-1}\mathbf{B}^{\mathrm{T}} \\ \mathbf{B} & \mathbf{0} & \mathbf{0} \\ \mathbf{Z}_t + \gamma'\mathbf{B}\mathbf{P}^{-1} & \mathbf{0} & \sigma\mathbf{B}\mathbf{P}^{-1}\mathbf{B}^{\mathrm{T}} \end{bmatrix} . \tag{D.2}$$

Using Eq. (74), Eqs. (69)–(73) follow as

$$\boldsymbol{\theta}_{n+1}^{j+1} = \boldsymbol{\theta}_{n+1}^{j} + \mathbf{T}^{-1}(\boldsymbol{\theta}_{n+1}^{j})\Delta\boldsymbol{\theta}^{j+1} , \tag{D.3}$$

$$\mathbf{q}_{n+1}^{j+1} = \boldsymbol{\tau}(\mathbf{q}_n, \boldsymbol{\theta}_{n+1}^{j+1}) , \tag{D.4}$$

$$\mathbf{v}_{n+1}^{j+1} = \mathbf{v}_{n+1}^{j} + \mathbf{P}^{-1}(\boldsymbol{\theta}_{n+1}^{j+1})\left(\gamma'\Delta\boldsymbol{\theta}^{j+1} + \sigma\mathbf{B}(\mathbf{q}_n)^{\mathrm{T}}\Delta\boldsymbol{\eta}\right) , \tag{D.5}$$

$$\dot{\mathbf{v}}_{n+1}^{j+1} = \dot{\mathbf{v}}_{n+1}^{j} + \mathbf{P}^{-1}(\boldsymbol{\theta}_{n+1}^{j+1})\left(\beta'\Delta\boldsymbol{\theta}^{j+1} + \epsilon'\sigma\mathbf{B}(\mathbf{q}_n)^{\mathrm{T}}\Delta\boldsymbol{\eta}\right) , \tag{D.6}$$

$$\lambda_{n+1}^{j+1} = \lambda_{n+1}^{j} + \Delta\lambda^{j+1} , \tag{D.7}$$

where

$$\epsilon' = \frac{1-\alpha_m}{h\gamma(1-\alpha_f)} . \tag{D.8}$$

A simple variant of our stabilized index-2 σ-Lie group generalized-α method follows for $\sigma = 1$. In this case, the matrix $\mathbf{P}^{-1}$ Eq. (65) is reduced to the identity matrix, and the dependence of the Jacobian Eq. (D.2) and Eqs. (D.3)–(D.6) on the local coordinates is waived and a correction step without local coordinates can be obtained, cf. Section 6.4. Alg. 3 outlines the computation of a single step using the proposed stabilized index-2 Lie group generalized-α method for $\sigma = 1$.

Algorithm 3 Numerical algorithm computing a Newton update for the proposed stabilized index-2 formulation for $\sigma = 1$.

```
function SolveTimeStepSigma1GGL($\mathbf{q}_n$, $\mathbf{v}_n$, $\dot{\mathbf{v}}_n$, $\mathbf{a}_n$)
 1: $\dot{\mathbf{v}}_{n+1} := \mathbf{0}$
 2: $\lambda_{n+1} := \mathbf{0}$
 3: $\mathbf{a}_{n+1} := (\alpha_f \dot{\mathbf{v}}_n - \alpha_m \mathbf{a}_n)/(1-\alpha_m)$
 4: $\mathbf{v}_{n+1} := \mathbf{v}_n + h(1-\gamma)\mathbf{a}_n + h\gamma\mathbf{a}_{n+1}$
 5: $\boldsymbol{\theta}_{n+1} := h\mathbf{v}_n + h^2\left(\frac{1}{2}-\beta\right)\mathbf{a}_n + h^2\beta\mathbf{a}_{n+1}$
 6: $\boldsymbol{\theta}_{n+1} := \left(\mathbf{I} - \frac{h\beta}{\gamma}\widehat{\mathbf{v}}_{n+1}\right)\boldsymbol{\theta}_{n+1}$
 7: $\mathbf{q}_{n+1} := \boldsymbol{\tau}(\mathbf{q}_n, \boldsymbol{\theta}_{n+1})$
 8: for $j = 1$ to $j_{max}$ do
 9:     $\mathbf{res} = \begin{bmatrix} \mathbf{r}(\mathbf{q}_{n+1}, \mathbf{v}_{n+1}, \dot{\mathbf{v}}_{n+1}, \lambda_{n+1}, t_{n+1}) \\ \boldsymbol{\Phi}(\mathbf{q}_{n+1}) \\ \dot{\boldsymbol{\Phi}}(\mathbf{q}_{n+1}, \mathbf{v}_{n+1}) \end{bmatrix}$
10:     if $\|\mathbf{res}\| < tol$ then
11:         break
12:     else
13:         $\mathbf{S}_t := \mathbf{S}_t(\mathbf{q}_{n+1}, \mathbf{v}_{n+1}, \dot{\mathbf{v}}_{n+1}, \lambda_{n+1}, t_{n+1})$
14:         $\begin{bmatrix} \Delta_{\boldsymbol{\theta}} \\ \Delta_\lambda \\ \Delta_\eta \end{bmatrix} := -\mathbf{S}_t^{-1}\,\mathbf{res}$
15:         $\mathbf{q}_{n+1} := \boldsymbol{\tau}(\mathbf{q}_{n+1}, \Delta_{\boldsymbol{\theta}})$
16:         $\mathbf{v}_{n+1} := \mathbf{v}_{n+1} + \gamma'\Delta_{\boldsymbol{\theta}} + \mathbf{B}(\mathbf{q}_n)^{\mathrm{T}}\Delta_{\boldsymbol{\eta}}$
17:         $\dot{\mathbf{v}}_{n+1} := \dot{\mathbf{v}}_{n+1} + \beta'\Delta_{\boldsymbol{\theta}} + \epsilon'\mathbf{B}(\mathbf{q}_n)^{\mathrm{T}}\Delta_{\boldsymbol{\eta}}$
18:         $\lambda_{n+1} := \lambda_{n+1} + \Delta_\lambda$
19:     end if
20: end for
21: $\mathbf{a}_{n+1} := \mathbf{a}_{n+1} + ((1-\alpha_f)/(1-\alpha_m))\dot{\mathbf{v}}_{n+1}$
22: return $\mathbf{q}_{n+1}$, $\mathbf{v}_{n+1}$, $\dot{\mathbf{v}}_{n+1}$, $\mathbf{a}_{n+1}$, $\lambda_{n+1}$,
end function
```

Data availability

Data will be made available on request.

References

[1] J. Chung, G.M. Hulbert, A time integration algorithm for structural dynamics with improved numerical dissipation: the generalized-α method, J. Appl. Mech. 60 (1993) 371, http://dx.doi.org/10.1115/1.2900803.

[2] M. Arnold, O. Brüls, Convergence of the generalized-α scheme for constrained mechanical systems, Multibody Syst. Dyn. 18 (2007) 185–202, http://dx.doi.org/10.1007/s11044-007-9084-0.

[3] O. Brüls, M. Arnold, The generalized-α scheme as a linear multistep integrator: Towards a general mechatronic simulator, J. Comput. Nonlinear Dyn. 3 (2008) 041007, http://dx.doi.org/10.1115/1.2960475.

[4] C. Lunk, B. Simeon, Solving constrained mechanical systems by the family of newmark and α-methods, 86, 2006, pp. 772–784, http://dx.doi.org/10.1002/zamm.200610285,

[5] L.O. Jay, B.C. Merwine, H.S.H. Yamashita, A two-stage extension of the generalized-α method for constrained systems in mechanics, in: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, 16th MSNDC, Virtual, Online, 2020, p. V002T02A006, http://dx.doi.org/10.1115/DETC2020-22385.

[6] O. Brüls, V. Acary, A. Cardona, Simultaneous enforcement of constraints at position and velocity levels in the nonsmooth generalized-α scheme, Comput. Methods Appl. Mech. Engrg. 281 (2014) 131–161, http://dx.doi.org/10.1016/j.cma.2014.07.025.

[7] A. Cosimo, J. Galvez, F.J. Cavalieri, A. Cardona, O. Brüls, A robust nonsmooth generalized-α scheme for flexible systems with impacts, Multibody Syst. Dyn. 48 (2020) 127–149, http://dx.doi.org/10.1007/s11044-019-09692-2.

[8] A. Müller, Singularity-free Lie group integration and geometrically consistent evaluation of multibody system models described in terms of standard absolute coordinates, J. Comput. Nonlinear Dyn. 17 (2022) 1–7, http://dx.doi.org/10.1115/1.4053368.

[9] M. Géradin, A. Cardona, Flexible Multibody Dynamics: A Finite Element Approach, John Wiley and Sons, New York, ISBN: 978-0-471-48990-0, 2001, p. 344.

[10] P. Singla, D. Mortari, J.L. Junkins, How to avoid singularity when using Euler angles? Adv. Astronaut. Sci. 119 (2005) 1409–1426.

[11] A.A. Shabana, Computational Dynamics, third ed., Wiley, 2010, p. 534, http://dx.doi.org/10.1002/9780470686850.

[12] O. Brüls, A. Cardona, M. Arnold, Lie group generalized-α time integration of constrained flexible multibody systems, Mech. Mach. Theory 48 (2012) 121–137, http://dx.doi.org/10.1016/j.mechmachtheory.2011.07.017.

[13] S. Holzinger, M. Arnold, J. Gerstmayr, Evaluation and implementation of Lie group integration methods for rigid multibody systems, Multibody Syst. Dyn. 62 (2023) 273–306, http://dx.doi.org/10.1007/s11044-024-09970-8.

[14] M. Arnold, A. Cardona, O. Brüls, A Lie algebra approach to Lie group time integration of constrained systems, in: P. Betsch (Ed.), Structure-Preserving Integrators in Nonlinear Structural Dynamics and Flexible Multibody Dynamics, Springer, Cham, 2016, pp. 91–158, http://dx.doi.org/10.1007/978-3-319-31879-0_3.
[15] A. Iserles, H. Munthe-Kaas, S. Nørsett, A. Zanna, Lie-group methods, Acta Numer. 9 (2000) 215–365, http://dx.doi.org/10.1017/S0962492900002154.
[16] J. Park, W.K. Chung, Geometric integration on euclidean group with application to articulated multibody systems, IEEE Trans. Robot. 21 (2005) 850–863, http://dx.doi.org/10.1109/TRO.2005.852253.
[17] J. Todesco, O. Brüls, Highly accurate differentiation of the exponential map and its tangent operator, Mech. Mach. Theory 190 (2023) http://dx.doi.org/10.1016/j.mechmachtheory.2023.105451.
[18] S. Faltinsen, A. Marthinsen, H.Z. Munthe-Kaas, Multistep methods integrating ordinary differential equations on manifolds, Appl. Numer. Math. 39 (2001) 349–365, http://dx.doi.org/10.1016/S0168-9274(01)00103-9.
[19] A. Cardona, M. Geradin, Time integration of the equations of motion in mechanism analysis, Comput. Struct. 33 (1989) 801–820, http://dx.doi.org/10.1016/0045-7949(89)90255-1.
[20] O. Brüls, A. Cardona, On the use of Lie group time integrators in multibody dynamics, J. Comput. Nonlinear Dyn. 5 (2010) 1–13, http://dx.doi.org/10.1115/1.4001370.
[21] J. Mäkinen, Critical study of newmark-scheme on manifold of finite rotations, Comput. Methods Appl. Mech. Engrg. 191 (2001) 817–828, http://dx.doi.org/10.1016/S0045-7825(01)00291-2.
[22] O. Brüls, M. Arnold, A. Cardona, Two Lie group formulations for dynamic multibody systems with large rotations, in: Proceedings of IDETC/MSNDC 2011, ASME 2011 International Design Engineering Technical Conferences, Washington, USA, 2011, http://dx.doi.org/10.1115/DETC2011-48132.
[23] V. Sonneville, O. Brüls, A formulation on the special euclidean group for dynamic analysis of multibody systems, J. Comput. Nonlinear Dyn. 9 (2014) 1–8, http://dx.doi.org/10.1115/1.4026569.
[24] V. Sonneville, A. Cardona, O. Brüls, Geometrically exact beam finite element formulated on the special euclidean group SE(3), Comput. Methods Appl. Mech. Engrg. 268 (2014) 451–474, http://dx.doi.org/10.1016/j.cma.2013.10.008.
[25] A. Müller, Coordinate mappings for rigid body motions, J. Comput. Nonlinear Dyn. 12 (2017) http://dx.doi.org/10.1115/1.4034730.
[26] M.A. Köbis, M. Arnold, Convergence of generalized-α time integration for nonlinear systems with stiff potential forces, Multibody Syst. Dyn. 37 (2016) 107–125, http://dx.doi.org/10.1007/s11044-015-9495-2.
[27] L. Jay, D. Negrut, A second order extension of the generalized-α method for constrained systems in mechanics, in: C. Bottasso (Ed.), Multibody Dynamics. Computational Methods and Applications, in: Computational Methods in Applied Sciences, vol. 12, Springer, Dordrecht, 2008, pp. 143–158, http://dx.doi.org/10.1007/978-1-4020-8829-2_8.
[28] A. Müller, Approximation of finite rigid body motions from velocity fields, ZAMM - J. Appl. Math. Mech. / Zeitschrift Für Angew. Math. Und Mech. 90 (2010) 514–521, http://dx.doi.org/10.1002/zamm.200900383.
[29] C.L. Bottasso, D. Dopico, L. Trainelli, On the optimal scaling of index three DAEs in multibody dynamics, Multibody Syst. Dyn. 19 (2008) 3–20, http://dx.doi.org/10.1007/s11044-007-9051-9.
[30] S. Holzinger, Computational Methods for Multibody Systems with Non-Redundant Parametrizations of Rigid Body Motion (Ph.D. thesis), University of Innsbruck, Innsbruck, 2023.
[31] J. Gerstmayr, Exudyn – A C++ based python package for flexible multibody systems, Multibody Syst. Dyn. 60 (2024) 533–561, http://dx.doi.org/10.1007/s11044-023-09937-1.
[32] L.F. Shampine, Evaluation of implicit formulas for the solution of ODEs, Bit 19 (1979) 495–502, http://dx.doi.org/10.1007/BF01931266.
[33] Z. Terze, A. Müller, D. Zlatar, Singularity-free time integration of rotational quaternions using non-redundant ordinary differential equations, Multibody Syst. Dyn. 38 (2016) 201–225, http://dx.doi.org/10.1007/s11044-016-9518-7.
[34] IFToM.M. Technical Committee for Multibody Dynamics, Library of computational benchmark problems, 2015.
[35] R. Masoudi, Spatial rigid slider-crank mechanism. Taken from "library of computational benchmark problems", 2022.
[36] A. Zwölfer, J. Gerstmayr, A concise nodal-based derivation of the floating frame of reference formulation for displacement-based solid finite elements: Avoiding inertia shape integrals, Multibody Syst. Dyn. 49 (2020) 291–313, http://dx.doi.org/10.1007/s11044-019-09716-x.
[37] A. Zwölfer, J. Gerstmayr, The nodal-based floating frame of reference formulation with modal reduction, Acta Mech. 232 (2021) 835–851, http://dx.doi.org/10.1007/s00707-020-02886-2.
[38] S. Holzinger, J. Gerstmayr, Time integration of rigid bodies modelled with three rotation parameters, Multibody Syst. Dyn. 53 (2021) 345–378, http://dx.doi.org/10.1007/s11044-021-09778-w.