Multibody Syst Dyn (2015) 34:275–305
DOI 10.1007/s11044-014-9439-2

Lie-group integration method for constrained multibody systems in state space

Zdravko Terze · Andreas Müller · Dario Zlatar

Received: 3 December 2013 / Accepted: 21 October 2014 / Published online: 22 November 2014
© Springer Science+Business Media Dordrecht 2014

Abstract Coordinate-free Lie-group integration method of arbitrary (and possibly higher) order of accuracy for constrained multibody systems (MBS) is proposed in the paper. Mathematical model of MBS dynamics is shaped as a DAE system of equations of index 1, whereas dynamics is evolving on the system state space modeled as a Lie-group. Since the formulated integration algorithm operates directly on the system manifold via MBS elements' angular velocities and rotational matrices, no local rotational coordinates are necessary, and kinematical differential equations (that are prone to singularities in the case of three-parameter-based local description of the rotational kinematics) are completely avoided. Basis of the integration procedure is the Munthe–Kaas algorithm for ODE integration on Lie-groups, which is reformulated and expanded to be applicable for the integration of constrained MBS in the DAE-index-1 form. In order to eliminate numerical constraint violation for generalized positions and velocities during the integration procedure, a constraint stabilization projection method based on constrained least-square minimization algorithm is introduced. Two numerical examples, heavy top dynamics and satellite with mounted 5-DOF manipulator, are presented. The proposed Lie-group DAE-index-1 integration scheme is easy-to-use for an MBS with kinematical constraints of general type, and it is especially suitable for dynamics of mechanical systems with large 3D rotations where standard (vector space) formulations might be inefficient due to kinematical singularities (three-parameter-based rotational coordinates) or additional kinematical constraints (redundant quaternion formulations).

Z. Terze (✉) · D. Zlatar
Department of Aeronautical Engineering, Faculty of Mechanical Engineering and Naval Architecture, University of Zagreb, Ivana Lučića 5, 10000 Zagreb, Croatia
e-mail: zdravko.terze@fsb.hr

D. Zlatar
e-mail: dario.zlatar@fsb.hr

A. Müller
Institute of Robotics, JKU Johannes Kepler University, 4040 Linz, Austria
e-mail: andreas.mueller@ieee.org

Keywords Lie-groups · Multibody systems dynamics · Numerical integration methods · DAE systems · Constraint violation stabilization · Munthe–Kaas integration algorithm · Special orthogonal group $SO(3)$

1 Introduction

Mathematical modelling and numerical simulation of large 3D motion and computational treatment of kinematical constraints are central problems in computational multibody dynamics. Unlike problems that belong to the 'classical' structural dynamics, which focused primarily on small displacements leading to linear problems, multibody dynamical models had to deal with kinematical constraints, large rotations and geometrical nonlinearities from the very beginning of this discipline of modern applied mechanics [1].

However, since large 3D rotations are not vectorial quantities [2], their representation within computational procedures and computer codes are far from trivial. Actually, a mathematical representation of 3D rotations and subsequent dynamical modelling, which are strongly related to a chosen representation, are characteristic points distinguishing one multibody computational procedure and integration algorithm from another. The standard 3D large rotation decomposition via 3-axis rotation angles (such as Euler and Tait–Bryan angles) has played an important role in the history of mechanical modelling [3, 4] (as well as in related disciplines, such as flight mechanics) and has also been adopted within multibody formalisms, especially for modelling cases where 3D rotations are large but with a limited domain definition.

The fact that 3D rotation permits its representation via three parameters, such as Euler angles, comes from its mathematical structure [3]: 3D rotation is described via rotation matrix $\mathbf{R}$ that belongs to the special orthogonal group $SO(3)$, which is also a manifold, that is, it is a nonlinear space that can be locally parameterized by coordinates that form a 'chart' [5, 6]. Along this line, Euler angles form a chart on $SO(3)$, providing a possibility to shift the calculations from the original nonlinear rotation manifold to a linear parameter space where standard vector-based algorithms can be used. However, as noted earlier, such parameterization has only local character, with the consequence that there inevitably exist singular positions, regardless of which local three-parameter coordinate system is used [7]. Nevertheless, any such three parameters constitute a minimal set of independent coordinates.

To avoid the problem of singularities, several authors proposed a descriptor form of the rotation description, introducing more (redundant) parameters than the DOF of the problem at hand (i.e. by introducing more than three parameters for a single 3D rotation). The most frequently used descriptor methodology calls for the introduction of a quaternion (set of four parameters) [4, 8], but some authors have also utilized nine-parameter descriptions based on the rotation matrix $\mathbf{R}$ rows and columns entities [9–11]. Although the descriptor formulation circumvents the singularity problem of local charts of the rotation manifold, it introduces more kinematical variables than necessary, and, by introducing the additional algebraic constraints due to redundancy of the parameters, it unnecessary complicates the modelling and numerical procedure.

In this paper, we will adopt a different route for the large 3D rotation description. Instead of using a local parameterization (such as Euler angles) with singularities or a redundant descriptor vector space formulation, we will solve the rotation part of a multibody system (MBS) dynamics directly on the rotation manifold of each rigid-body configuration space. By following this route, a local parameterization of each body rotation as well as kinematical differential equations (differential equations that relate angular velocities and derivatives

of the introduced rotation parameters and which are, in the case of three parameters model description, prone to numerical singularities) are completely avoided in the standard form. To this end, the proposed computational procedure operates only with the body rotation matrices $\mathbf{R}$ and angular velocities $\boldsymbol{\omega}$, avoiding thus local rotation parameters (therefore some authors call this formulation as 'coordinate-free'), which leads to a more compact and numerically more efficient integration routine.

However, working directly on nonlinear manifolds assumes that the integration algorithm is designed appropriately. In particular, vector-space operations of addition and linear combination that are intrinsic to all standard integration ODE/DAE routines (such as Runge–Kutta schemes, Adams–Moulton, Newmark, generalized α-method, BDF algorithm, etc.) are no longer valid. Instead of using vector-space methods, ODE integration methods that operate on nonlinear manifolds (a vector space can be understood as a linear manifold) and Lie-groups (specially structured manifolds that also possess mathematical structure of the group) are described in the literature and ready to be used [12, 13]. However, these routines are almost exclusively designed to be used for ODE problems on manifolds only, meaning that kinematic (algebraic) constraints issues and equivalent DAE problems (that are central to MBS applications) cannot be solved straightforwardly by the existing 'off-the-shelf' Lie-group procedures.

The goal of the paper is to propose a mathematical model and a numerical integration algorithm based on the Lie-group integration scheme (in particular, Munthe–Kaas Lie-group integration scheme [13, 14]) that will be pertinent to the constrained multibody dynamics. To this end, the state space of an MBS is modelled as a Lie-group, and the mathematical model of the numerical scheme is cast as DAE of index 1. The method primarily focuses on the dynamics of a rigid-body MBS, although its application can also be easily extended on the systems that possess elastic components (a numerical procedure focused on the integration of the flexible MBS, designed in the Lie-group settings in index 3 DAE form, is described in [15]).

Novel contribution of the paper can be summarized as follows:

(1) For the first time, mathematical formulation of MBS state space, introduced as Lie-group, has been described.
(2) The novel numerical algorithm that operates in a state space Lie-group has been proposed.

    The algorithm is based on the Munthe–Kaas numerical procedure for solving Lie-group ODE problems. However, in this paper, the Munthe–Kaas approach has been expanded to be pertinent for DAE index 1 problems in Lie-groups and applied for constrained MBS.
(3) A novel constraint violation stabilization method that operates directly in an MBS state space Lie-group has been proposed.

By discussing the novel points described above, the paper proposes a compact integration procedure for the constrained MBS on Lie-groups. The procedure can be of arbitrary order of accuracy, including higher orders, such as 4th-order of accuracy Lie-group integration scheme that is demonstrated within the presented examples (actually, for the first time, to our best knowledge, an MBS Lie-group integration scheme is proposed that can be easily implemented as a higher-order method). Since the proposed method operates directly with the MBS elements' angular velocities and rotational matrices (meaning that local parameterization is avoided), this formulation circumvents the well-known problems of the classical vector-space-based MBS integration procedures, such as the kinematic singularity of the three-parameter rotation basis, re-parameterization of the system kinematics during integration, and numerical non-efficiency of the kinematic differential equations.

Here it should be emphasized, for completeness, that geometry of the rotation space, modelled as a Lie-group, was already studied and applied within the geometrically consistent integration schemes described in the pioneering works [16–18]. Also, the integration methods of problems of evolution in the rotation group based on generalization of Runge–Kutta methods that are similar to the Munthe–Kaas approach are discussed in [19].

2 Configuration space and state space of unconstrained MBS as Lie-groups

The configuration space of an unconstrained MBS comprising k bodies is modelled as a Lie-group $\mathcal{G} = \mathcal{R}^3 \times SO(3) \times \cdots \times \mathcal{R}^3 \times SO(3)$ (k copies of $\mathcal{R}^3 \times SO(3)$) with elements in the form

$$q = (\mathbf{r}_1, \mathbf{R}_1, \ldots, \mathbf{r}_k, \mathbf{R}_k), \tag{1}$$

where $SO(3) = \{\mathbf{R} \in \mathcal{R}^{3\times 3} : \mathbf{R}\mathbf{R}^T = \mathbf{I}, \det \mathbf{R} = +1\}$ is the three-dimensional special orthogonal group (Lie-group of rigid body rotation). Each factor $\mathcal{R}^3 \times SO(3)$ represents a configuration of a single rigid body i represented by $(\mathbf{r}_i, \mathbf{R}_i)$, its mass centre global position vector and body rotation matrix with regard to global frame. $\mathcal{G}$ is a Lie-group of dimension $n = 6k$, where k is the number of rigid bodies. The left multiplication in the group is given as $L_q : \mathcal{G} \to \mathcal{G}, \bar{q} \mapsto q \cdot \bar{q}$, where the product operation on $\mathcal{G}$ is defined by $q \cdot \bar{q} = (\mathbf{r}_1 + \bar{\mathbf{r}}_1, \mathbf{R}_1\bar{\mathbf{R}}_1, \ldots, \mathbf{r}_k + \bar{\mathbf{r}}_k, \mathbf{R}_k\bar{\mathbf{R}}_k)$, and the group identity element is $e = (\mathbf{0}_1, \mathbf{I}_1, \ldots, \mathbf{0}_k, \mathbf{I}_k)$.

The angular velocity of a rigid body expressed in the coordinate system attached to the body is given by the left-invariant vector field $\tilde{\boldsymbol{\omega}}_i \in so(3)$ defined as

$$\dot{\mathbf{R}}_i(t) = \mathbf{R}_i(t)\tilde{\boldsymbol{\omega}}_i, \tag{2}$$

with $so(3)$ being the Lie-algebra of $SO(3)$. The element of the Lie-algebra $\tilde{\boldsymbol{\omega}}_i \in so(3)$ can be identified with $\mathcal{R}^3$ via mapping operator, which maps a vector $\boldsymbol{\omega}_i \in \mathcal{R}^3$ to a skew-symmetric matrix $\tilde{\boldsymbol{\omega}}_i \in so(3)$ [3]. Therefore, the velocity of one body can thus be represented by the couple $(\mathbf{v}_i, \tilde{\boldsymbol{\omega}}_i) \in \mathcal{R}^3 \times so(3)$ or $(\mathbf{v}_i, \boldsymbol{\omega}_i) \in \mathcal{R}^3 \times \mathcal{R}^3$. With $\mathcal{G}$ so defined, its Lie-algebra is given as $g = \mathcal{R}^3 \times so(3) \times \cdots \times \mathcal{R}^3 \times so(3)$ (k copies of $\mathcal{R}^3 \times so(3)$) with elements of the form

$$v = (\mathbf{v}_1, \tilde{\boldsymbol{\omega}}_1, \ldots, \mathbf{v}_k, \tilde{\boldsymbol{\omega}}_k). \tag{3}$$

Alternatively, the vector notation $\mathbf{v} = [\mathbf{v}_1^T, \boldsymbol{\omega}_1^T, \ldots, \mathbf{v}_k^T, \boldsymbol{\omega}_k^T]^T$ is used for formulation of mathematical models for practical (matrix-notation-based) computations, see Sects. 4 and 7.

Aiming for the application of the Lie-group integration scheme, the MBS state space must also be expressed as a Lie-group. With the above, the MBS state space is introduced as $\mathcal{S} = \mathcal{G} \times g$, that is, $\mathcal{S} = \mathcal{R}^3 \times SO(3) \times \cdots \times \mathcal{R}^3 \times SO(3) \times \mathcal{R}^3 \times so(3) \times \cdots \times \mathcal{R}^3 \times so(3) \cong T\mathcal{G}$ with the elements

$$x = (\mathbf{r}_1, \mathbf{R}_1, \ldots, \mathbf{r}_k, \mathbf{R}_k, \mathbf{v}_1, \tilde{\boldsymbol{\omega}}_1, \ldots, \mathbf{v}_k, \tilde{\boldsymbol{\omega}}_k). \tag{4}$$

$\mathcal{S}$ is the left-trivialization of the tangent bundle $T\mathcal{G}$. This is a Lie-group itself that possesses the Lie-algebra $s = \mathcal{R}^3 \times so(3) \times \cdots \times \mathcal{R}^3 \times so(3) \times \mathcal{R}^3 \times \mathcal{R}^3 \times \cdots \times \mathcal{R}^3 \times \mathcal{R}^3$ with the element

$$z = (\mathbf{v}_1, \tilde{\boldsymbol{\omega}}_1, \ldots, \mathbf{v}_k, \tilde{\boldsymbol{\omega}}_k, \dot{\mathbf{v}}_1, \dot{\boldsymbol{\omega}}_1, \ldots, \dot{\mathbf{v}}_k, \dot{\boldsymbol{\omega}}_k). \tag{5}$$

The element $z \in s$ summarizes the MBS velocity and acceleration. Considering a single body for the sake of clarity, the operations on the state-space Lie-group $\mathcal{S}$ and its Lie-algebra s are introduced as follows:

(a) Left multiplication in $\mathcal{S} = \mathcal{R}^3 \times SO(3) \times \mathcal{R}^3 \times so(3)$ is defined as $L_x : \mathcal{S} \to \mathcal{S}, \bar{x} \mapsto x \cdot \bar{x}$, where the product operation is given as

$$(a, b, c, d) \cdot (e, f, g, h) = (a + e, b \cdot f, c + g, d + h).$$

(b) Addition in $s = \mathcal{R}^3 \times so(3) \times \mathcal{R}^3 \times \mathcal{R}^3$:

$$(u, w, c, d) + (\bar{u}, \bar{w}, \bar{c}, \bar{d}) = (u + \bar{u}, w + \bar{w}, c + \bar{c}, d + \bar{d}).$$

(c) Multiplication by scalar in $s = \mathcal{R}^3 \times so(3) \times \mathcal{R}^3 \times \mathcal{R}^3 : \alpha(u, w, c, d) = (\alpha u, \alpha w, \alpha c, \alpha d)$.

(d) Exponential map in $s = \mathcal{R}^3 \times so(3) \times \mathcal{R}^3 \times \mathcal{R}^3 : \exp(u, w, c, d) = (u, \text{expm}(w), c, d)$.

(e) Bracket in $s = \mathcal{R}^3 \times so(3) \times \mathcal{R}^3 \times \mathcal{R}^3 : [(u, w, c, d), (\bar{u}, \bar{w}, \bar{c}, \bar{d})] = (0, [w, \bar{w}], 0, 0)$.

On the right-hand side of these definitions, '$\cdot$' is the multiplication in $SO(3)$, '$+$' is the addition in $\mathcal{R}^3$ and $so(3)$, 'expm' is the exponential map on $so(3)$, and $[w, \bar{w}]$ is a Lie bracket (matrix commutator $[w, \bar{w}] = w\bar{w} - \bar{w}w$ for the matrix Lie-algebra $so(3)$). For an MBS consisting of k bodies, these operations are component-wise carried over to $\mathcal{S}$ and s.

It will be necessary to relate the tangent space of the configuration space Lie-group $\mathcal{G}$ at the given configuration q to its Lie-algebra. This is given by the differential (tangent map) of the left multiplication map L_q as

$$L'_q : T_e\mathcal{G} \to T_q\mathcal{G} : (\mathbf{v}_1, \tilde{\boldsymbol{\omega}}_1, \ldots, \mathbf{v}_k, \tilde{\boldsymbol{\omega}}_k) \mapsto (\dot{\mathbf{r}}_1, \mathbf{R}_1\tilde{\boldsymbol{\omega}}_1, \ldots, \dot{\mathbf{r}}_k, \mathbf{R}_k\tilde{\boldsymbol{\omega}}_k), \tag{6}$$

which is the identity map for translation velocity, that is, $\mathbf{v}_i = \dot{\mathbf{r}}_i$. It relates the time derivative $\dot{q}$ of the MBS configuration to the MBS velocity $v \in \mathcal{g}$ via the tangent map as

$$\dot{q} = L'_q(v). \tag{7}$$

This is the extension of the left-invariant Poisson equation (2), relating $\dot{\mathbf{R}}_i$ and $\tilde{\boldsymbol{\omega}}_i$, to a system of k unconstrained rigid bodies when $\mathcal{R}^3 \times SO(3)$ is used as the configuration space of a rigid body. Or, speaking prose all our lives, this is merely the collection of $\dot{\mathbf{r}}_i = \mathbf{v}_\text{i}$ and $\dot{\mathbf{R}}_i(t) = \mathbf{R}_i(t)\tilde{\boldsymbol{\omega}}_i$ for all bodies $i = 1, \ldots, k$, relating translational and angular velocities to the time derivatives of position vector and rotation matrix when $\mathcal{R}^3 \times SO(3)$ is used as a rigid-body configuration space.

Analogously, on the state space $\mathcal{S} = \mathcal{G} \times \mathcal{g}$ the differential of the left multiplication map, given as

$$\begin{aligned} &L'_x : T_e\mathcal{S} \to T_x\mathcal{S}, \\ &(\mathbf{v}_1, \tilde{\boldsymbol{\omega}}_1, \ldots, \mathbf{v}_k, \tilde{\boldsymbol{\omega}}_k, \dot{\mathbf{v}}_1, \dot{\boldsymbol{\omega}}_1, \ldots, \dot{\mathbf{v}}_k, \dot{\boldsymbol{\omega}}_k) \mapsto (\dot{\mathbf{r}}_1, \mathbf{R}_1\tilde{\boldsymbol{\omega}}_1, \ldots, \dot{\mathbf{r}}_k, \mathbf{R}_k\tilde{\boldsymbol{\omega}}_k, \dot{\mathbf{v}}_1, \dot{\boldsymbol{\omega}}_1, \ldots, \dot{\mathbf{v}}_k, \dot{\boldsymbol{\omega}}_k), \end{aligned} \tag{8}$$

relates the Lie-algebra s to the tangent space at x. This allows relating the time derivative of the MBS state to the velocity and acceleration of the unconstrained MBS, summarized in $z \in s$, as

$$\dot{x} = L'_x(z). \tag{9}$$

In summary, the Lie-groups $\mathcal{G}$ and $\mathcal{S}$ serve as configuration and state space of the unconstrained MBS. Relation (7) yields kinematic reconstruction equations that express the MBS configuration (generalized positions) change as the system is moving with a certain velocity. By solving them on the basis of the known velocity field, one determines the motion of the MBS. Now, the kinematic relation (9) determines the change of the state of an unconstrained MBS as it moves with certain velocity and acceleration. This is the central relation for the following Lie-group integration scheme.

3 Kinematic reconstruction of MBS motion

The attitude of a rotating rigid body $\mathbf{R}(t)$ can be reconstructed from its angular velocity $\tilde{\boldsymbol{\omega}}_i(t) \in so(3)$ by solving (2). Starting from an initial rotation $\mathbf{R}_0$, the solution for the rotation matrix is given as $\mathbf{R}(t) = \mathbf{R}_0 \exp(\tilde{\mathbf{u}}(t))$ [12, 20], where the closed form of the exponential mapping on $SO(3)$ is given by the Rodrigues formula [7], and $\mathbf{u}(t) \in \mathcal{R}^3$ is the instantaneous rotation vector.

The kinematic reconstruction equation (2) is an ordinary differential equation (ODE) on the Lie-group $SO(3)$. It can be solved numerically by using geometric integration methods designed to operate in Lie-groups [12, 13]. Integration of attitude kinematics (2) is a classical application of the Lie-group integration schemes, which can be extended to unconstrained MBS. As shown above, this gives rise to the kinematic reconstruction equation (7) that allows for direct application of a Lie-group integration scheme to the kinematics of the overall MBS. However, this is not central topic of this paper.

This paper aims to describe a unified approach for numerical integration of kinematics and dynamics of MBS. To this end, the crucial observation is that Eqs. (7) on the Lie-group configuration space and Eqs. (9) on the Lie-group state space have the same mathematical structure, allowing for integration within the same geometric procedure. However, the MBS motion must also satisfy kinematic constraints the MBS is subjected to. To this end, the DAE index 1 formulation will be incorporated into the Lie-group integration framework, ensuring that the velocities and accelerations in $z \in s$ on the right-hand side of (9) satisfy the imposed kinematic constraints. To be precise, the DAE index 1 formulation allows for immanent acceleration constraints satisfaction only, whereas velocities (as well as configuration coordinates) must generally be corrected by constraint stabilization procedure as described in Sect. 6.

In short, the basic underlying idea of the geometric MBS integration method introduced in this paper is to solve (9) with a Lie-group integration scheme, whereas the $z \in s$ is to be determined by the MBS dynamics, consistently with the kinematic constraints imposed to the system.

4 Dynamics formulation for constrained MBS in Lie-group setting

To formulate a dynamical model of the system, we start from the equations of motion (based on Newton–Euler equations) of constrained MBS introduced in the form

$$\mathbf{M}\dot{\mathbf{v}} + \mathbf{C}^{\mathrm{T}}(q)\boldsymbol{\lambda} = \mathbf{Q}(q, \mathbf{v}, t), \tag{10a}$$

$$\dot{q} = L'_q(v), \tag{10b}$$

$$\boldsymbol{\Phi}(q) = 0, \tag{10c}$$

where $\mathbf{M}$ is a constant $n \times n$-dimensional system inertia matrix composed of masses and inertia tensors of the unconstrained system bodies (whose configuration space is $\mathcal{G}$, $n = 6k$, assuming k bodies), $\mathbf{v} \in \mathcal{R}^n$, $\mathbf{v} = [\mathbf{v}_1^T, \boldsymbol{\omega}_1^T, \ldots, \mathbf{v}_k^T, \boldsymbol{\omega}_k^T]^T$ are system velocities in usual matrix notation, q (system configuration) and v (system velocities with angular velocities expressed as $\tilde{\boldsymbol{\omega}}_i \in so(3)$) are given by Eqs. (1) and (3), $\mathbf{Q}$ represents external and all other forces, $\boldsymbol{\lambda} \in \mathcal{R}^m$ is the vector of Lagrange multipliers, and $\mathbf{C}$ is the $m \times n$-dimensional constraint Jacobian, so that $\boldsymbol{\Phi}'(v) = \mathbf{C}(q)\mathbf{v}$, where $\boldsymbol{\Phi}'$ is the differential of the scleronomic constraint mapping $\boldsymbol{\Phi}(q) : \mathcal{G} \to \mathcal{R}^m$. The latter imposes geometric (configuration) constraints of the MBS on the configuration space $\mathcal{G}$. Consequently, the MBS is constrained to evolve on the $(n-m)$-dimensional sub-manifold $\mathcal{N} = \{q \in \mathcal{G} : \boldsymbol{\Phi}(q) = \mathbf{0}\}$. Some authors call the configuration space $\mathcal{G}$ (space of unconstrained bodies $\mathcal{G}$ where geometric constraints $\boldsymbol{\Phi}(q) = \mathbf{0}$ are defined) an ambient configuration space [21].

System (10a)–(10c) is a DAE system of index 3 on the Lie-group $\mathcal{G}$. The matrix equation (10a) represents dynamical equations of the MBS subjected to the configuration constraints (10c) that are complemented by the kinematic reconstruction equations (10b) on $\mathcal{G}$. As it is well-known, the integration results of (10a)–(10c) should also satisfy system kinematical constraints on the velocity

$$\mathbf{C}(q)\mathbf{v} = \mathbf{0}, \tag{11}$$

and the acceleration level

$$\mathbf{C}(q)\dot{\mathbf{v}} = \boldsymbol{\xi}(q, \mathbf{v}), \tag{12}$$

which are obtained by once and twice differentiation of the configuration constraints $\boldsymbol{\Phi}(q) = \mathbf{0}$ (the same procedure as it is in the case of classical MBS formulations). Here, we assumed that the MBS is subjected to scleronomic constraints, but the model is easily extended to incorporate rheonomic constraints as well. As shown below, the constraint equations in the matrix notation (10c), (11) and (12) as well as the dynamical Eq. (10a) have identical expressions as they would have if the dynamical model had been formulated by using classical MBS formulations [1, 4, 22].

Although system (10a)–(10c) can be integrated directly by using DAE index 3 numerical schemes [15], in order to formulate dynamical model to be pertinent to utilization within the system kinematic reconstruction on $\mathcal{S} = \mathcal{G} \times \mathcal{g}$ via Munthe–Kaas type of ODE integrators, we reformulate (10a)–(10c) to the DAE index 1 form that yields

$$\begin{bmatrix} \mathbf{M} & \mathbf{C}^{\mathrm{T}} \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{v}} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{Q} \\ \boldsymbol{\xi} \end{bmatrix}. \tag{13}$$

Above, during the well-known system index reduction procedure, the acceleration constraint equation (12) was included in (13) instead of the position constraint equation (10c), whereas the system dynamical equations (10a) remains the same as it appears in (10a)–(10c). As it is well-known from the classical MBS literature [4, 22, 23] and research papers in the field (see, for example, [24] and references cited there or [25]), the integration results of (13) should also satisfy kinematical constraints on the position and velocity level (10c) and (11) (so-called 'hidden constraints'), beside the constraints at the acceleration level (12) that will be automatically satisfied during integration since they are explicitly included in dynamical model (13).

The concept of the Lie-group state space model, which is a core of the MBS geometric integration algorithm presented in the paper, is based on the utilization of (9) and deduction of $z(x, t) \in s$ from the constrained MBS dynamical model (13). To this end, for the given

generalized forces $\mathbf{Q}(x,t)$, the linear algebraic system (13) can be solved giving rise to a mapping

$$\dot{\mathbf{v}} = \boldsymbol{\mathcal{H}}(t, x, \mathbf{Q}). \tag{14}$$

By introducing the mapping $\boldsymbol{\mathcal{F}} : \boldsymbol{\mathcal{R}} \times \mathcal{S} \to s$ defined as

$$\boldsymbol{\mathcal{F}}(t,x) = \big(v, \boldsymbol{\mathcal{H}}(t,x)\big), \tag{15}$$

the MBS dynamics equation yields with (9) the state-space form as

$$\dot{x} = L'_x\big(\boldsymbol{\mathcal{F}}(t,x)\big). \tag{16}$$

Here it should be emphasized that the index 1 DAE (13) is exactly in the same (matrix) form as it appears in the classical vector-space-based MBS formulations [4], assuming that the MBS velocity and acceleration fields are expressed via $\mathbf{v}$ and $\dot{\mathbf{v}}$ (i.e. via bodies' (absolute) translational and angular velocities and accelerations) and kinematical constraints (10c), (11) and (12) are derived on the basis of the system configuration expressed via q (i.e. via bodies' position vectors $\mathbf{r}_i$ and rotation (body orientation) matrices $\mathbf{R}_i$) and $\mathbf{v}$ (or v) and $\dot{\mathbf{v}}$. Speaking of technical details, in [26] one can find principles of derivation of kinematical constraints of all lower-pair joints and some higher-pair joints on the basis of such an approach. See also Appendix B and examples presented in Sect. 7. For example, spherical joint kinematical constraints that define heavy top kinematical structure (Sect. 7.1) are given by (25), (26) and (27), whereas the pertinent dynamical equations and system Jacobian $\mathbf{C}$ are expressed by (23), (24) and (28). In matrix notation given here, these expressions are just the same as if they would appear in the classical MBS formulations, and such is also the heavy top final dynamical model that yields DAE index 1 form given by (29).

What is different, however, is the integration procedure. On the contrary to the classical MBS formulations that include reconstruction of the system kinematics based on the local (singularity-prone) rotation coordinates [4, 27], the Lie-group geometric algorithm presented in the sequel allows for the straightforward integration of DAE index 1 dynamical model (13) together with the simultaneous MBS kinematic reconstruction expressed directly in the coordinate-free form by means of q. Usual kinematic singularities are thus completely avoided, and the overall integration algorithm takes particularly compact form.

5 Lie-group integration algorithm for constrained MBS

Equation (16) is a first-order ODE on the Lie-group $\mathcal{S}$ that can be solved numerically using geometric integration methods operating directly on Lie-groups. In this paper, the Munthe–Kaas Lie-group method [12–14] is adopted as the underlying integration algorithm. By following this approach, the solution of (16) is expressed in the form

$$x(t) = x(0)\exp\big(u(t)\big), \tag{17}$$

where $u(t) \in s$ is a solution of the ODE system in the Lie-algebra s [12, 13]

$$\dot{u} = \operatorname{dexp}^{-1}_{-u}\big(z(x,t)\big), \quad u(0) = 0, \tag{18}$$

and the operator $\operatorname{dexp}^{-1}_{-u}$ is introduced in (A.9). Since (18) is an ODE defined in the Lie-algebra s that is linear space, any vector-space ODE integrator, such as the 4th-order Runge–Kutta (RK) method, can be used for its integration. This is the focal point in the Munthe–Kaas (MK) type of integrators: instead of working in a nonlinear manifold where (16) is

defined, the integration point is transferred into a local tangent (vector) space by (18) and integrated in a Lie-algebra by using standard ODE integration methods, see Fig. 1. In this paper, the RK scheme is used for integration in a Lie-algebra, as it was the case in the original Munthe–Kaas Lie-group algorithm.

Now, the Lie-algebra substitution equation (18) can be written in the form

$$\dot{u} = \mathrm{dexp}_{-u}^{-1}\big(\boldsymbol{\mathcal{F}}(t,x)\big), \quad u(0) = 0, \tag{19}$$

and the standard RK scheme can be applied to solve it within each integration step of the Lie-group MK-RK algorithm.

Therefore, by following [12, 14, 28], the integration step $\bar{n}$ in the time interval $t \in [t_{\bar{n}-1}, t_{\bar{n}} = t_{\bar{n}-1} + h]$ of the MK-RK algorithm can be given in the form

$$\begin{aligned}
&x_0 = x_{\bar{n}-1}\\
&\quad \text{for } i = 1, 2, \ldots, s\\
&u_i = h \sum_{j=1}^{i-1} a_{ij} f_j^*\\
&f_i = \boldsymbol{\mathcal{F}}\big(t_{\bar{n}-1} + c_i h, x_0 \exp(u_i)\big)\\
&f_i^* = \mathrm{dexp}_{-u_i}^{-1}(f_i, n)\\
&\quad \text{end}\\
&w_{\bar{n}} = h \sum_{j=1}^{s} b_j f_j^*\\
&x_{\bar{n}} = x_0 \exp(w_{\bar{n}})
\end{aligned} \tag{20}$$

where u, f_i, f_i^*, $w_{\bar{n}} \in s$, the coefficients a_{ij}, b_j, c_i are given by the s-stage nth-order Runge–Kutta method's Butcher table [12], and the function $\mathrm{dexp}_{-u_i}^{-1}(f_i, n)$ is the truncated form of (A.9), where the upper summation in (A.4) is specified as n (to keep the accuracy of the overall MK-RK algorithm in accordance with the accuracy of the chosen nth-order RK scheme [13]). Closed forms for fast computations of the functions exp and dexp^{-1} for $SO(3)$ Lie-group can be found in Appendix B in [12].

The outline of the integration procedure is depicted in Fig. 1. Once accelerations of the system are obtained for the current integration step by solving the linear algebraic system of the classical (matrix-notation-based) DAE index 1 dynamical model of constrained MBS (13), an integration point is 'lifted' from the (nonlinear) state space group manifold $\mathcal{S}$ to its local (vector) tangent space (i.e. Lie-algebra s), where it is integrated as a classical vector space ODE problem by means of the new local (tangent space) coordinates u (as indicated in (19), the initial condition for the step ODE integration is always $u(0) = 0$). Once Lie-algebra ODE integration is completed, the step integration point is 'pulled-back' to the state space manifold $\mathcal{S}$ by the exponential function, which completes the proposed method integration step.

6 Constraint violation stabilization procedure

The numerical solution obtained by the described algorithm will automatically satisfy the constraint equation at the acceleration level (12) since this equation is explicitly incorporated

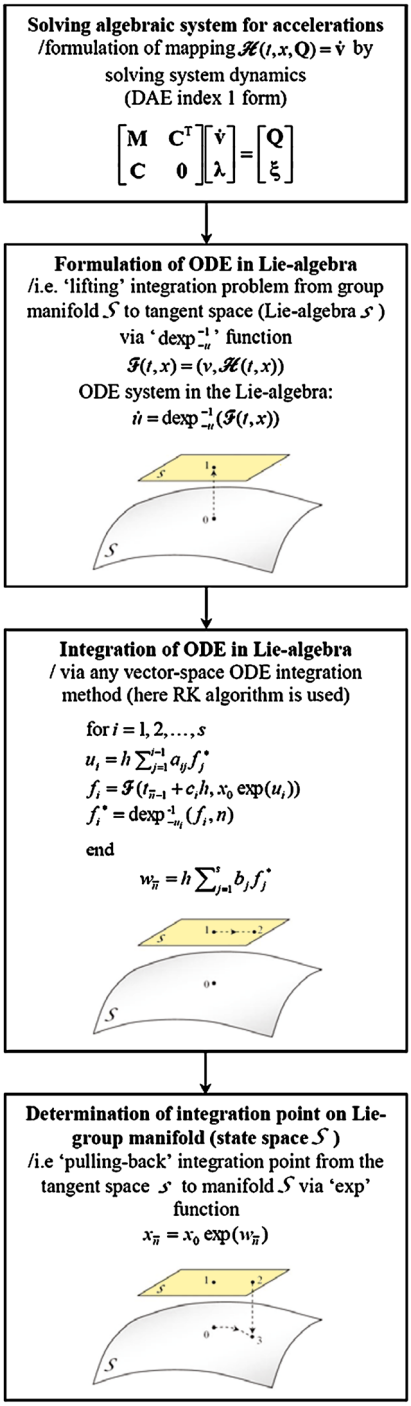


Fig. 1 Outline of constrained MBS Lie-group state space integration step

in the DAE index 1 system (13) and the mapping $\mathcal{H}$ in (14). However, similarly to the case with standard vector-space-based formulations [1, 4, 22], constraint equations for generalized positions (10c) and velocities (11) will be inevitably violated during the straightforward integration based on the DAE index 1 system. To tackle this problem, a constraint stabilization procedure at the position and velocity level has to be incorporated into the integration procedure, as it is common practice in MBS dynamics [1, 25, 29].

For the purpose of the constraint stabilization procedure that operates directly on $\mathcal{S}$ (that is MBS state space formulated as a Lie-group where the proposed geometric integrator returns unstabilized integration results), we propose a projective method based on the constrained least-square problem given in the form

$$\min_{(q_i,\mathbf{v}_i)} \left\| \begin{pmatrix} q_i - \hat{q}_i \\ \mathbf{v}_i - \hat{\mathbf{v}}_i \end{pmatrix} \right\|_{\mathbf{W}}^2, \tag{21}$$

$$\mathbf{\Phi}(q) = 0, \qquad \mathbf{R}_i \mathbf{R}_i^T = \mathbf{I}, \qquad \dot{\mathbf{\Phi}}(q, \mathbf{v}) = 0, \tag{22}$$

(where $\| \ \|_{\mathbf{W}}$ denotes the weighted norm, and the difference operation in $q_i - \hat{q}_i$ stands for the difference between the corresponding vector-column $\mathbf{r}_i$ and matrix $\mathbf{R}_i$ elements in q_i and $\hat{q}_i$, i.e. it is not a difference between the group elements, which is an operation that is not defined). During the minimization procedure (21), the projected variables have to satisfy the constraint equations (22) to obtain the stabilized values $q_i, \mathbf{v}_i$ for each rigid body i (i running from 1 to k; k is the number of rigid bodies), whereas $\hat{q}_i, \hat{\mathbf{v}}_i$ are the unstabilized values obtained from the integrator for the current integration step. It should be emphasized here that the constraint violation corrections imposed by (21) and (22) are actually introduced in the $\mathcal{R}^{12k}$ linear space where $\mathcal{S}$ is embedded. Hence, the equations $\mathbf{R}_i \mathbf{R}_i^T = \mathbf{I}$ are included in the projection algorithm to make sure that the stabilization procedure returns results in $\mathcal{S}$ and does not undermine the orthogonality of $\mathbf{R}_i$, which is geometrically preserved by the presented Lie-group integration algorithm (that respects a nonlinear state space manifold and returns unstabilized integration results in $\mathcal{S}$).

Within the stabilization procedure, after each integration step on the Lie-group $\mathcal{S}$, integration values are adjusted to be in accordance with the constraints $\mathbf{\Phi}(q) = 0$ and $\dot{\mathbf{\Phi}}(q, \mathbf{v}) = 0$ by preserving the orthogonality of $\mathbf{R}_i$ during the process (as explained, we treat $\mathbf{R}_i$ as it is valid $\mathbf{R}_i \in GL(3)$ and impose the orthogonality equations $\mathbf{R}_i \mathbf{R}_i^T = \mathbf{I}$ as 'external' condition during minimization). In (21), the unit matrix weighted norm $\mathbf{I}$ was used for the stabilization of the configuration q, whereas for the velocities $\mathbf{v}$, two weighted norms were used and compared in the examples presented in the next section: the unity norm $\mathbf{I}$ and the system inertia matrix $\mathbf{M}$. As pointed out in [30], basically any positive-definite matrix qualifies for this selection, and some authors also proposed modified solutions with the aim of improving the numerical efficiency or energy performance of the algorithm (see [30] and references there).

Technically, a numerical solution of the projection step can be computed iteratively using the Gauss–Newton algorithm, which is essentially based on the generalized inverses (or pseudo-inverse) of the system constraint matrix and which represents a well-known common procedure in the domain of numerical solving of algebraic systems [31, 32]. Also, different approaches, such as the penalty method or augmented Lagrange method [33] as well as constraint violation stabilizations using projections based on optimal coordinates partitioning [25, 34] and constraint manifold orthogonal directions [29], are proposed in the literature (see also Chap. VII.2 in [35]). However, those algorithms are designed to operate primarily within the classical (vector-space-based) MBS formulations.

Another constraint violation stabilization procedure that is specialized to operate on Lie-groups within the MBS applications is proposed in [36]. This algorithm uses local exponential coordinates in order to introduce constraint violation corrections around the current un-stabilized integration values in $\mathcal{S}$. By being based on Lie-group correction updates, this method respects inherent geometry of $\mathcal{S}$, meaning that the orthogonality condition $\mathbf{R}_i\mathbf{R}_i^T = \mathbf{I}$ must not be explicitly imposed during the process.

As a final remark on the proposed stabilization procedure, it can be said that, although the algorithm expressed by (21) and (22) is based on the corrections in linear $\mathcal{R}^{12k}$ where $\mathcal{S}$ is embedded, due to the implemented orthogonality condition $\mathbf{R}_i\mathbf{R}_i^T = \mathbf{I}$, it returns corrected integration values 'directly' in $\mathcal{S}$. Hence, this procedure is fully complementary to the proposed Lie-group geometric integrator. Moreover, the structure of the presented stabilization algorithm is similar to the standard least square projection routines that are very well researched (and widely used) within the classical MBS formulations [31, 37]. Based on this, it can be expected that potential users of the Lie-group integrator presented in this paper should not have technical difficulties to accommodate standard linear space MBS stabilization routines to operate on $\mathcal{S}$ by utilising (21) and (22).

Speaking of numerical performance, the stabilization algorithm generally performs very well: it reaches stabilized solution (with a chosen accuracy) within very few iteration steps, see presented examples. Since the Lie-group integrator returns $\mathbf{R}_i \in SO(3)$ that satisfy condition $\mathbf{R}_i\mathbf{R}_i^T = \mathbf{I}$ at 'machine' precision (and theses values are initial conditions for the subsequent stabilization procedure for the given integration step), it is to be expected that the imposed minimization constraint $\mathbf{R}_i\mathbf{R}_i^T = \mathbf{I}$ (which assures that the stabilized solution also belongs to $\mathcal{S}$) should not rise numerical difficulties. In other words, the additional numerical effort of projection of the stabilized solution (calculated in $\mathcal{R}^{12k}$) back to $\mathcal{S}$ should not be a significant one. Moreover, it can be expected that in some cases the whole procedure can be even more numerically efficient than the local coordinates Lie-group stabilization algorithm [36], which assures stabilized results to be 'automatically' consistent with a nonlinear geometry of $\mathcal{S}$. This method is based solely on Lie-group operational updates that respect inherent geometry of the manifold $\mathcal{S}$, but incorporation of local correction coordinates (as well as additional exponential map projection) requires also numerical effort.

7 Examples

7.1 Heavy top

As a numerical illustration, the dynamics of a heavy top (see Fig. 2) is presented in the first example. The top is modelled as a constrained mechanical system that leads to a DAE formulation, and the equations that govern the system dynamics and kinematics are presented as follows.

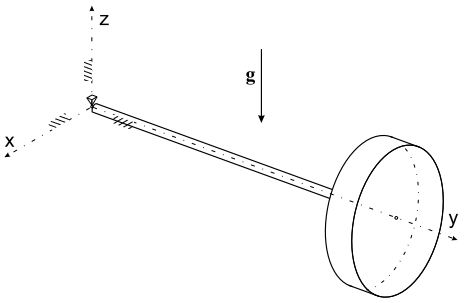
The constrained Newton–Euler equations governing the system dynamics can be given in the form

$$m\dot{\mathbf{v}}_C - \boldsymbol{\lambda} = m\mathbf{g}, \tag{23}$$

$$\mathbf{J}\dot{\boldsymbol{\omega}} + \tilde{\mathbf{r}}_b\mathbf{R}^T\boldsymbol{\lambda} = -\tilde{\boldsymbol{\omega}}\mathbf{J}\boldsymbol{\omega}, \tag{24}$$

where $\boldsymbol{\omega}$ represents the body's angular velocity, $\mathbf{v}_C$ is velocity of the body's mass centre, m and $\mathbf{J}$ are the body mass and tensor of inertia, $\boldsymbol{\lambda}$ stands for the joint reaction forces, $\mathbf{g}$ is the gravity vector, $\mathbf{r}_b$ is the (constant) body mass centre measured in the body-fixed frame,

Fig. 2 Heavy top



and $\mathbf{R} \in SO(3)$ is the rotation matrix that relates the body frame to the inertial frame. By introducing $\mathbf{r}$ as the body mass centre position, the mechanical system constraints at the position, velocity and acceleration level due to the spherical joint are given as

$$-\mathbf{r} + \mathbf{R}\mathbf{r}_b = \mathbf{0}, \tag{25}$$

$$[-\mathbf{I}_3 \quad -\mathbf{R}\tilde{\mathbf{r}}_b] \begin{bmatrix} \mathbf{v}_C \\ \boldsymbol{\omega} \end{bmatrix} = \mathbf{0}, \tag{26}$$

$$[-\mathbf{I}_3 \quad -\mathbf{R}\tilde{\mathbf{r}}_b] \begin{bmatrix} \dot{\mathbf{v}}_C \\ \dot{\boldsymbol{\omega}} \end{bmatrix} = -\mathbf{R}\tilde{\boldsymbol{\omega}}\tilde{\boldsymbol{\omega}}\mathbf{r}_b, \tag{27}$$

and the system constraint Jacobian, denoted by $\mathbf{C}$, is

$$\mathbf{C} = [-\mathbf{I}_3 \quad -\mathbf{R}\tilde{\mathbf{r}}_b], \tag{28}$$

which allows for assembling the equations of the system dynamics in DAE of index 1 form

$$\begin{bmatrix} m\mathbf{I}_3 & \mathbf{0} & -\mathbf{I}_3 \\ \mathbf{0} & \mathbf{J} & \tilde{\mathbf{r}}_b\mathbf{R}^T \\ -\mathbf{I}_3 & -\mathbf{R}\tilde{\mathbf{r}}_b & \mathbf{0} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{v}}_C \\ \dot{\boldsymbol{\omega}} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} m\mathbf{g} \\ -\tilde{\boldsymbol{\omega}}\mathbf{J}\boldsymbol{\omega} \\ -\mathbf{R}\tilde{\boldsymbol{\omega}}\tilde{\boldsymbol{\omega}}\mathbf{r}_b \end{bmatrix}. \tag{29}$$

Equation (29) comprises Eqs. (23), (24) and (27) and has the same formal shape as (13), where $\mathbf{C}$ is given by (28), and the system generalized inertia matrix is formulated as

$$\mathbf{M} = \begin{bmatrix} m\mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{J} \end{bmatrix}, \tag{30}$$

whereas the system velocities $\mathbf{v} \in \mathcal{R}^6$ are expressed as $\mathbf{v} = [\mathbf{v}_C^T, \boldsymbol{\omega}^T]^T$. During the numerical integration of (29), the integration results must also satisfy the position and velocity constraint equations (25) and (26). This will be accomplished by a constraint violation stabilization procedure. In the context of derivation of (26), expression (2) has been used as well as the relation $\tilde{\boldsymbol{\omega}}\mathbf{r}_b = -\tilde{\mathbf{r}}_b\boldsymbol{\omega}$.

In standard units, the value of the mass is set to $m = 15$, the inertia tensor is $\mathbf{J} = \text{diag}[0.234375, 0.46875, 0234375]$, the gravity vector is $\mathbf{g} = [0\ 0\ -9.81]^T$, the position of the centre of the mass is $\mathbf{r}_b = [0\ 1\ 0]^T$, and the initial conditions are $\mathbf{R}_0 = \mathbf{I}$ and $\boldsymbol{\omega}_0 = [0\ 150\ -4.61538]^T$. For the purpose of numerical integration, the proposed state space Lie-group integration procedure (based on the 4th-order MK-RK algorithm) was implemented in MATLAB, and the unstabilized integration results obtained with the fixed integration time step size $h = 1\text{e}{-4}$ are presented in Figs. 3–8. The mass centre position is

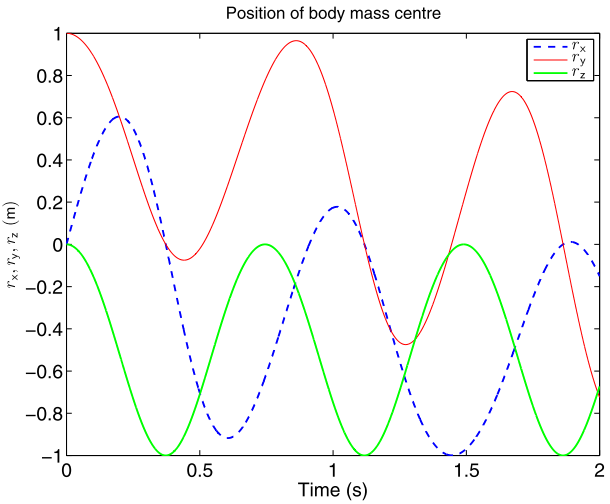


Fig. 3 Coordinates of body mass centre

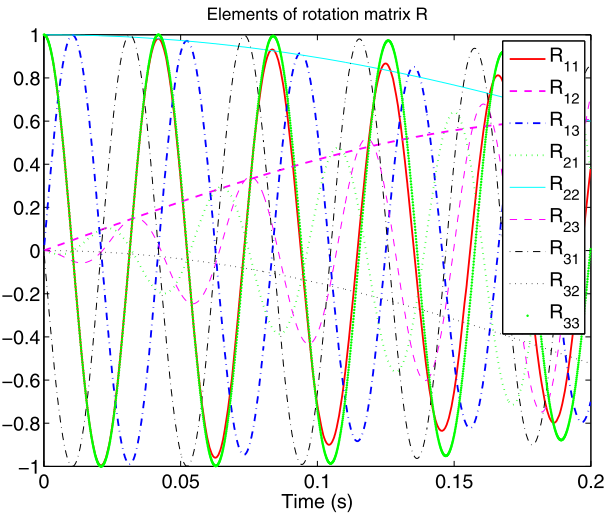


Fig. 4 Elements of rotation matrix **R**

shown in Fig. 3, whereas the elements of the body rotation matrix $\mathbf{R} \in SO(3)$ are given in Fig. 4.

Integrating the mechanical system directly on the manifold $\mathcal{S}$, as it is enabled by the proposed Lie-group integration method, allowed for avoidance of parameterization singularities, which would have occurred if local coordinates had been used for large 3D rigid body rotation parameterization.

Indeed, by inspecting the integral curves of the system position and rotation (Figs. 3 and 4), it is visible that all results are smooth functions without any discontinuities whatsoever. If, on the other hand, the integration procedure for large 3D rotation had been based on a set of three-parameter-based local coordinates (which would have led to a 'standard' vector space integration routine), then the discontinuities due to exceeding of the domain of definition of particular variables would have occurred.

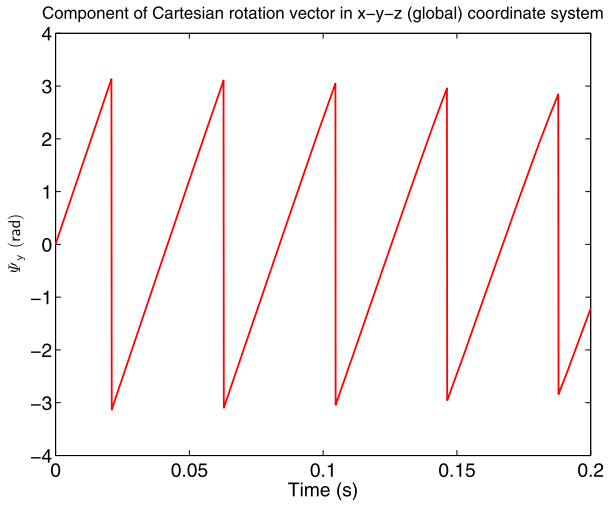


Fig. 5 Component Ψ_y of Cartesian rotation vector $\boldsymbol{\Psi}$, defined as $\|\boldsymbol{\Psi}\| \leq \pi$

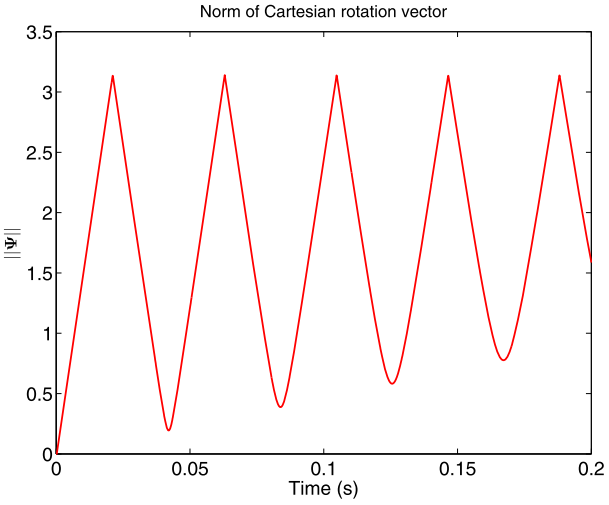


Fig. 6 Norm of Cartesian rotation vector

This is demonstrated here in Figs. 5, 6 and 7, where the same integration results as those depicted in Fig. 4 are presented via Cartesian rotation vector (by means of which the attitude (orientation) of the body is determined by the vector's unit-direction that determines an axis of the rotation, whereas the magnitude of the vector represents angular rotation around the axis [26]) and Tait–Bryan angles. Since the Cartesian vector 3D rotation parameterization is based on $SO(3)$ exponential mapping that reaches singularities after rotation of magnitude $2k\pi$, where k is an integer, singularity discontinuities for a multiple pivoting rigid body are clearly visible. Also, Fig. 7 shows the Tait–Bryan angles (the Euler angles defined as 3-2-1 successive rotations) of the same motion, where the domain definition singularities are also visible at time discontinuities of the angle ϕ. Discontinuities of this kind would call for re-parameterization of local coordinates, which generally leads to unnecessary complexities that slow down the integration and can be a source of potential numerical instability.

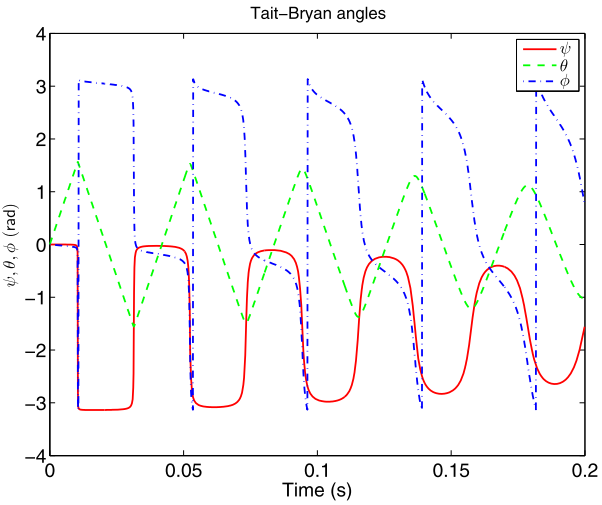


Fig. 7 Tait–Bryan angles

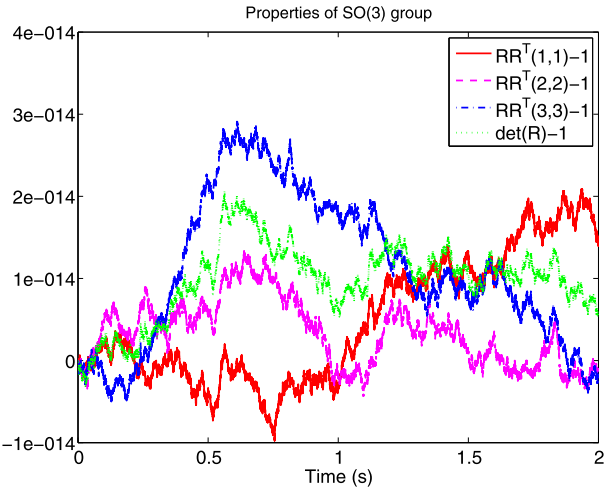


Fig. 8 Properties of rotation matrix $\mathbf{R} \in SO(3)$. Numerical errors of diagonal elements of product $\mathbf{R}\mathbf{R}^T = \mathbf{I}$ and determinant $\det \mathbf{R} = +1$

Furthermore, although the results presented in Figs. 3–8 are obtained by direct numerical integration without applying the constraint violation stabilization procedure, the orthogonality properties of the rotation matrix $\mathbf{R} \in SO(3)$ are preserved 'exactly' by the Lie-group-based integration procedure. This is shown in Fig. 8, presenting the difference between the matrix entries along the main diagonal of the matrix $\mathbf{R}\mathbf{R}^T$ and the identity matrix $\mathbf{I}$, as well as the errors of the matrix determinant $\det \mathbf{R} = +1$ compared to the analytical solution. It is visible here that the orthogonality of $\mathbf{R} \in SO(3)$ is preserved at excellent numerical tolerance even for the unstabilized numerical integration (and it would have been preserved even if a first-order Lie-group integrator had been applied). On the other hand, if we had attempted to solve a kinematical reconstruction equation (2) (which is also part of the established mathematical model) with a standard vector space integration routine, the orthogonality of the rotation matrix $\mathbf{R} \in SO(3)$ would have been lost after just a few integration steps for most of the applied standard integrators. This is so because the orthogonality of $\mathbf{R} \in SO(3)$

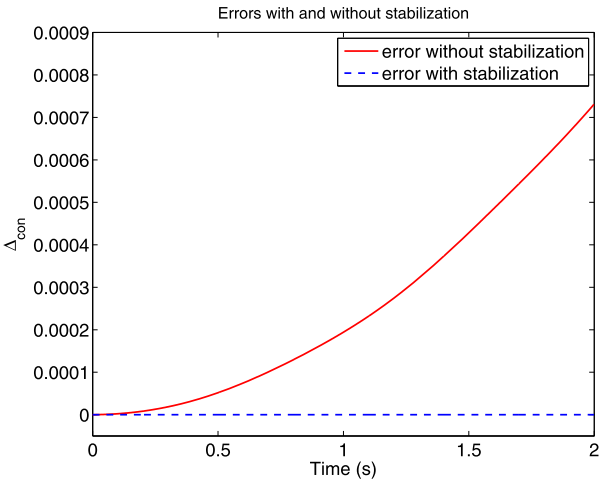


Fig. 9 Generalized position stabilization procedure. Errors with and without stabilization, $h = 1\text{e}{-3}$

is a configuration constraint of quadratic type that only few standard ODE integrators can successfully satisfy [13].

However, although the orthogonal properties of $\mathbf{R} \in SO(3)$ are preserved by the Lie-group integration method, the general configuration and velocity constraints that come into play because of the MBS kinematical constraints will be numerically violated during the DAE index 1 integration (see Sect. 6). Therefore, constraint violation stabilization should be performed within the integration procedure, and this is illustrated in the sequel, where the proposed stabilization method is applied and tested within the framework of the presented example.

In order to validate the characteristics of the stabilization algorithm, the relative constraint violation error

$$\boldsymbol{\Delta}_{\text{con}} = \frac{\|\boldsymbol{\theta}^{\text{int}} - \boldsymbol{\theta}^{\text{con}}\|}{\|\boldsymbol{\theta}^{\text{con}}\|}, \tag{31}$$

has been adopted as a performance measure within the framework of several numerical experiments. At the generalized position level, the variables $\boldsymbol{\theta}^{\text{int}}$ and $\boldsymbol{\theta}^{\text{con}}$ are introduced as $\boldsymbol{\theta}^{\text{int}} = \mathbf{r}$ and $\boldsymbol{\theta}^{\text{con}} = \mathbf{R}\mathbf{r}_b$, where $\mathbf{r}$ and $\mathbf{R}$ are obtained directly by the integrator. With the variables so introduced, $\boldsymbol{\theta}^{\text{int}} - \boldsymbol{\theta}^{\text{con}}$ of (31) basically represents the left-hand side of Eq. (25), meaning that the numerator of (31) yields the generalized position constraint violation (if it differs from zero).

For the generalized positions, (31) is computed with and without the implementation of the stabilization algorithm for different integration step sizes, and the results are presented in Figs. 9–11. In Fig. 9, the stabilized integration results are compared with the results obtained via straightforward unstabilized integration. When unstabilized integration was performed with the step lengths of $h = 1\text{e}{-4}$ and $h = 1\text{e}{-5}$, the error diagrams completely resembled the unstabilized error depicted in Fig. 9 but with cumulative errors 7.393e−8 and 7.359e−12, respectively, for the motion domain of 2 seconds.

The stabilized numerical results obtained by using integration time steps of length $h = 1\text{e}{-3}$ and $h = 1\text{e}{-4}$ are presented in Figs. 10 and 11. Here, it is visible that the stabilization algorithm delivers good and reliable stabilization results. The iterative calculation of a nonlinear optimization procedure based on (21) and (22) was implemented using the MATLAB routine *fmincon* for convenience, whereas a quasi-Newton-type algorithm can be used

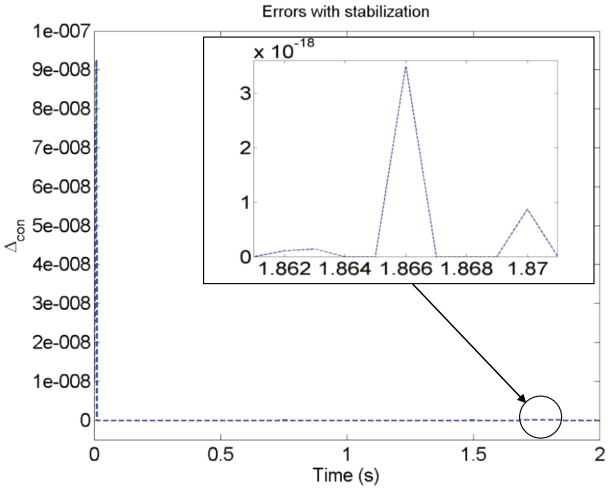


Fig. 10 Generalized position stabilization procedure, $h = 1e{-}3$, $Tol = 1e{-}7$

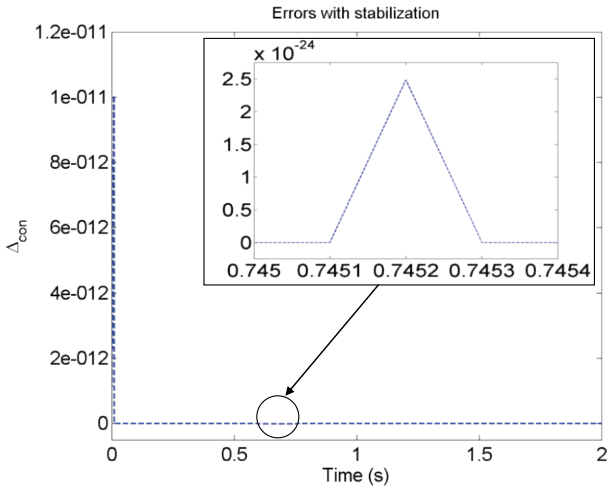


Fig. 11 Generalized position stabilization procedure, $h = 1e{-}4$, $Tol = 1e{-}11$

for the same purpose in general MBS codes. The stabilization procedure converged very quickly (see Fig. 15) with the constraint satisfaction *fmincon* tolerances set as $Tol = 1e{-}7$ and $Tol = 1e{-}11$ for the integration steps $h = 1e{-}3$ and $h = 1e{-}4$, respectively. In Figs. 10 and 11, it is visible that the obtained accuracies of constraint violation stabilization are completely controlled by the imposed numerical tolerance *Tol* that, within *fmincon* routine, controls the numerical satisfaction of the optimization constraint equations imposed by (22). Actually, the overall stabilization accuracy is equal to the one specified as the constraint satisfaction tolerance only at the beginning of the computational interval (Figs. 10 and 11), and later on, the process converges very quickly to much better stabilization results, although the overall stabilization computational cost was always minimal (convergence was achieved in two optimization steps at the most, see Fig. 15).

Expression (31) was also used for a validation of the constraint stabilization procedure at the velocity level. For this purpose, the variables $\boldsymbol{\theta}^{\text{int}}$ and $\boldsymbol{\theta}^{\text{con}}$ are introduced as $\boldsymbol{\theta}^{\text{int}} = -\mathbf{v}$ and $\boldsymbol{\theta}^{\text{con}} = \mathbf{R}\tilde{\mathbf{r}}_b\boldsymbol{\omega}$ (see Eq. (26)), meaning that the numerator of (31) yields the generalized

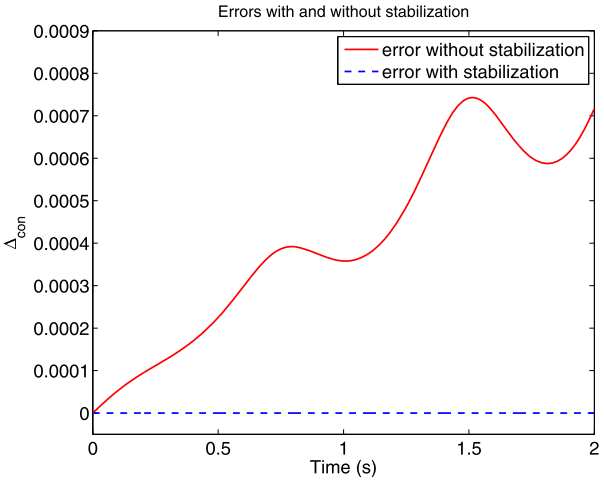


Fig. 12 Velocity stabilization procedure. Errors with and without stabilization, $h = 1\text{e}{-3}$

velocity constraint violation. With these definitions, expression (31) is computed with and without the implementation of the stabilization algorithm, and the results obtained with the fixed integration step $h = 1\text{e}{-3}$ are presented in Figs. 12–14. Here, a stabilization algorithm based on Eqs. (21) and (22) was applied, and the optimization tolerances within MATLAB routine *fmincon* were set as $Tol = 5\text{e}{-4}$ and $Tol = 1\text{e}{-7}$. Figures 13 and 14 show the differences in the stabilization performances when the norm weighted matrices in Eq. (21) are set as $\mathbf{W} = \mathbf{I}$ and $\mathbf{W} = \mathbf{M}$. It is visible that both choices return comparable results, with the weighted matrix $\mathbf{W} = \mathbf{M}$ performing slightly better (especially at the beginning of the simulation period in the case where lower tolerance was specified).

Similarly as for the positions, an unstabilized integration returned a steady growth of the constraint violation at the velocity level as well (see Fig. 12). When the unstabilized integration was performed with the steps $h = 1\text{e}{-4}$ and $h = 1\text{e}{-5}$, the error diagrams resembled the unstabilized error depicted in Fig. 12 but with cumulative errors $7.196\text{e}{-8}$ and $7.199\text{e}{-12}$ for the motion domain of 2 seconds.

Figure 15 shows the number of iterative steps within the stabilization optimization procedures at the position and velocity level that were needed to reach the specified tolerance for different integration steps and constraint satisfaction tolerances. It is visible that maximum two iteration steps were needed in all the analysed cases. At the position level, approximately the same numerical efforts were needed for both integration steps $h = 1\text{e}{-3}$ and $h = 1\text{e}{-4}$ because of the higher stabilization tolerance that was imposed on the shorter step ($Tol = 1\text{e}{-11}$ vs. $Tol = 1\text{e}{-7}$). On the other hand, at the velocity level, higher stabilization tolerance ($Tol = 1\text{e}{-7}$) resulted in a bigger numerical effort than needed for lower tolerance ($Tol = 5\text{e}{-4}$) in the same integration step ($h = 1\text{e}{-3}$).

7.2 Satellite with 5-DOF manipulator

The second illustrative numerical example is a cylindrically shaped satellite equipped with a manipulator consisting of three links connected with different types of joints. The kinematic structure of the system is presented in Fig. 16. The centre of mass of the satellite body is restricted from translation by a fixed-point constraint (see Appendix B for derivation of the algebraic constraint equations imposed by the fixed-point constraint; in the example presented in this section, it is set $\mathbf{r}_{b_1}^{\mathrm{FP}} = \mathbf{0}$, see Appendix B), and its rotational motion with

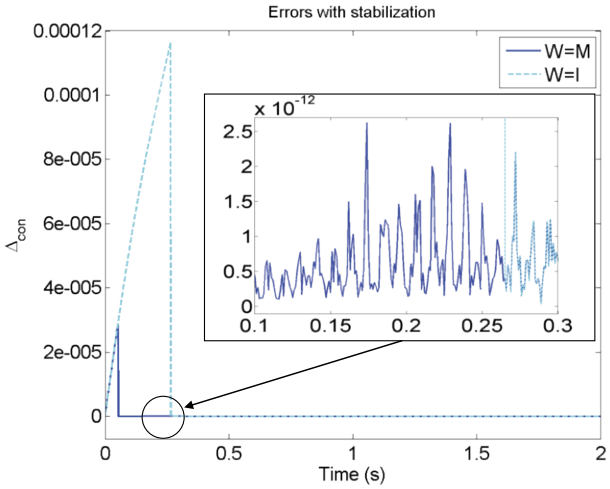


Fig. 13 Velocity stabilization procedure, $h = 1e{-}3$, $Tol = 5e{-}4$

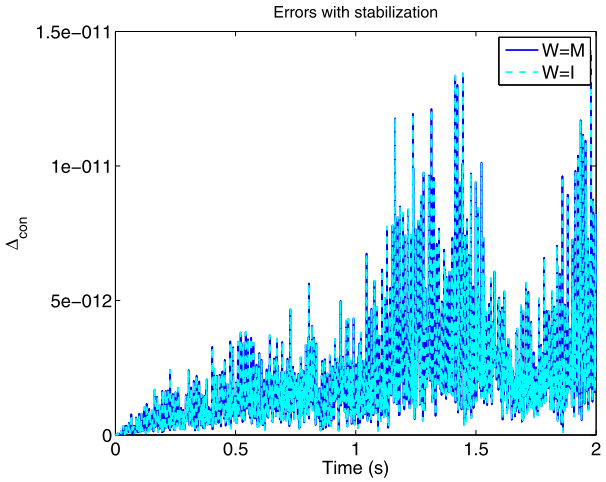


Fig. 14 Velocity stabilization procedure, $h = 1e{-}3$, $Tol = 1e{-}7$

constant angular velocity is prescribed by the rheonomic constraint given at the velocity level in the form $\mathbf{R}_1\boldsymbol{\omega}_1 - \boldsymbol{\omega}_{g1} = \mathbf{0}$ (where $\boldsymbol{\omega}_{g1} = const$ is the angular velocity of the satellite body in the global coordinate system, and $\mathbf{R}_1$ and $\boldsymbol{\omega}_1$ determine the satellite body rotation matrix and its angular velocity in the local (body) coordinate system, respectively).

The first manipulator element, the base rod, is connected with the satellite body via a spherical joint. On the other end, the base rod is connected via a revolute joint to the second manipulator element, the slider rod. The slider rod itself is connected by a prismatic joint with a tool, a slider, which is able to translate along the slider rod.

The system governing equations are formulated in the form of Eq. (13), where particular matrices pertinent to this example are given as follows.

The system accelerations vector $\dot{\mathbf{v}}$ is given by

$$\dot{\mathbf{v}} = \begin{bmatrix} \dot{\mathbf{v}}_1^T & \dot{\boldsymbol{\omega}}_1^T & \dot{\mathbf{v}}_2^T & \dot{\boldsymbol{\omega}}_2^T & \dot{\mathbf{v}}_3^T & \dot{\boldsymbol{\omega}}_3^T & \dot{\mathbf{v}}_4^T & \dot{\boldsymbol{\omega}}_4^T \end{bmatrix}^T, \tag{32}$$

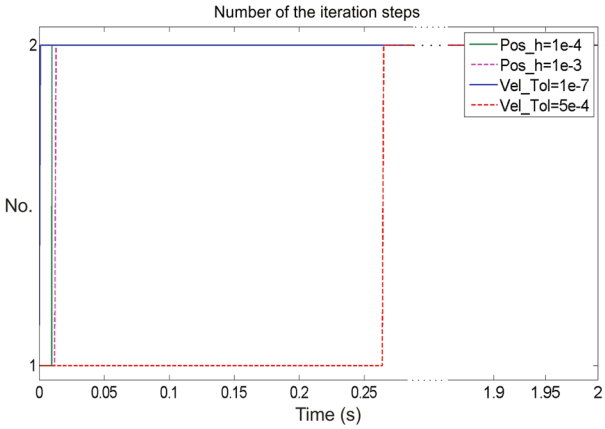


Fig. 15 Number of iteration steps within least square minimization procedure

where $\dot{\mathbf{v}}_i$ is the translational acceleration of the mass centre of the ith body, and $\dot{\boldsymbol{\omega}}_i$ is its angular acceleration, and the generalized inertia matrix of the system is given by

$$\mathbf{M} = \operatorname{diag}\left[m_1\mathbf{I}_3 \quad \mathbf{J}_1 \quad m_2\mathbf{I}_3 \quad \mathbf{J}_2 \quad m_3\mathbf{I}_3 \quad \mathbf{J}_3 \quad m_4\mathbf{I}_3 \quad \mathbf{J}_4\right]. \tag{33}$$

The global constraint matrix of the system is shaped by combining all joints and rheonomic constraint matrices in the form

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_1^{\mathrm{FP}(3\times6)} & & \mathbf{0}_2^{(3\times6)} & & \mathbf{0}_3^{(3\times6)} & & \mathbf{0}_4^{(3\times6)} \\ & \mathbf{C}_{1,2}^{\mathrm{SJ}(3\times12)} & & & \mathbf{0}_3^{(3\times6)} & & \mathbf{0}_4^{(3\times6)} \\ \mathbf{0}_1^{(5\times6)} & & & \mathbf{C}_{2,3}^{\mathrm{RJ}(5\times12)} & & & \mathbf{0}_4^{(5\times6)} \\ \mathbf{0}_1^{(5\times6)} & & \mathbf{0}_2^{(5\times6)} & & & \mathbf{C}_{3,4}^{\mathrm{PJ}(5\times12)} & \\ \mathbf{C}_1^{\mathrm{RH}(3\times6)} & & \mathbf{0}_2^{(3\times6)} & & \mathbf{0}_3^{(3\times6)} & & \mathbf{0}_4^{(3\times6)} \end{bmatrix}, \tag{34}$$

where $\mathbf{C}_1^{\mathrm{FP}(3\times6)}$ is the 3×6-dimensional fixed-point constraint matrix on the main satellite body (body 1, lower index 1), $\mathbf{C}_{1,2}^{\mathrm{SJ}(3\times12)}$ represents the 3×12-dimensional spherical joint constraint matrix that connects bodies 1 and 2, $\mathbf{C}_{2,3}^{\mathrm{RJ}(5\times12)}$ represents the 5×12-dimensional revolute joint constraint matrix on the bodies 2 and 3, $\mathbf{C}_{3,4}^{\mathrm{PJ}(5\times12)}$ represents the 5×12-dimensional prismatic constraint matrix that connects bodies 3 and 4, and $\mathbf{C}_1^{\mathrm{RH}(3\times6)}$ is the constraint matrix due to the rheonomic constraint $\mathbf{R}_1\boldsymbol{\omega}_1 - \boldsymbol{\omega}_{g1} = \mathbf{0}$ on the main satellite body. The right-hand-side vector $\boldsymbol{\xi}$ of the global acceleration constraints is formed as

$$\boldsymbol{\xi} = \left[\boldsymbol{\xi}_1^{\mathrm{FP}(3\times1)^T} \quad \boldsymbol{\xi}_{1,2}^{\mathrm{SJ}(3\times1)^T} \quad \boldsymbol{\xi}_{2,3}^{\mathrm{RJ}(5\times1)^T} \quad \boldsymbol{\xi}_{3,4}^{\mathrm{PJ}(5\times1)^T} \quad \boldsymbol{\xi}_1^{\mathrm{RH}(3\times1)^T}\right]^T, \tag{35}$$

where the upper indices determine the types of the constraints involved, and the lower indices stand for the numbers marking the constrained body.

For brevity, here we will only describe a derivation of the spherical joint constraint matrix and the right-hand-side term of the acceleration constraint, whereas the derivation of constraint matrices and the acceleration terms of other types of constraints follow accordingly (see details in [26] and Appendix B). For the spherical joint, three algebraic constraint equations at the generalized displacement level can be written as

$$\boldsymbol{\Phi}^{\mathrm{SJ}} = \mathbf{r}_1 + \mathbf{R}_1\mathbf{r}_{b_1}^{\mathrm{SJ}} - \mathbf{r}_2 - \mathbf{R}_2\mathbf{r}_{b_2}^{\mathrm{SJ}} = \mathbf{0}, \tag{36}$$

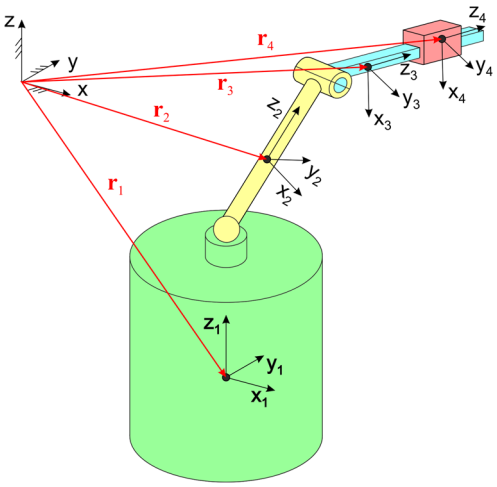


Fig. 16 Model of the satellite with mounted 5-DOF manipulator

where $\mathbf{r}_{b_1}^{\mathrm{SJ}}$ and $\mathbf{r}_{b_2}^{\mathrm{SJ}}$ represent the location of the spherical joint in the local coordinate systems of the bodies (the bodies whose kinematics is constrained by the joint, here bodies 1 and 2), $\mathbf{r}_1$ and $\mathbf{r}_2$ are the positions of the mass centres of the bodies in the global coordinate system (see Fig. 17), and $\mathbf{R}_1$ and $\mathbf{R}_2$ are the body rotation matrices.

Differentiating (36) once with respect to time, the constraint equation at the velocity level is obtained as

$$\dot{\mathbf{\Phi}}^{\mathrm{SJ}} = \mathbf{v}_1 + \mathbf{R}_1\tilde{\boldsymbol{\omega}}_1\mathbf{r}_{b_1}^{\mathrm{SJ}} - \mathbf{v}_2 - \mathbf{R}_2\tilde{\boldsymbol{\omega}}_2\mathbf{r}_{b_2}^{\mathrm{SJ}} = \mathbf{0}, \tag{37}$$

whereas another time differentiation gives the acceleration constraint in the form

$$\ddot{\mathbf{\Phi}}^{\mathrm{SJ}} = \dot{\mathbf{v}}_1 - \mathbf{R}_1\tilde{\mathbf{r}}_{b_1}^{\mathrm{SJ}}\dot{\boldsymbol{\omega}}_1 + \mathbf{R}_1\tilde{\boldsymbol{\omega}}_1\tilde{\boldsymbol{\omega}}_1\mathbf{r}_{b_1}^{\mathrm{SJ}} - \dot{\mathbf{v}}_2 + \mathbf{R}_2\tilde{\mathbf{r}}_{b_2}^{\mathrm{SJ}}\dot{\boldsymbol{\omega}}_2 - \mathbf{R}_2\tilde{\boldsymbol{\omega}}_2\tilde{\boldsymbol{\omega}}_2\mathbf{r}_{b_2}^{\mathrm{SJ}} = \mathbf{0}. \tag{38}$$

In (37) and (38), $\boldsymbol{\omega}_{1,2}$ and $\dot{\boldsymbol{\omega}}_{1,2}$ are the bodies' angular velocities and accelerations, respectively. To obtain the acceleration constraint in the form that can be included in the system dynamics governing equations (13), the acceleration constraint (38) can be written as

$$\mathbf{C}_{1,2}^{\mathrm{SJ}(3\times12)}\dot{\mathbf{v}}_{1,2}^{(12\times1)} = \boldsymbol{\xi}_{1,2}^{\mathrm{SJ}(3\times1)}, \tag{39}$$

where $\dot{\mathbf{v}}_{1,2}^{(12\times1)} = [\dot{\mathbf{v}}_1^T\ \dot{\boldsymbol{\omega}}_1^T\ \dot{\mathbf{v}}_2^T\ \dot{\boldsymbol{\omega}}_2^T]^T$ is the generalized acceleration vector of bodies 1 and 2, and the spherical joint constraint matrix $\mathbf{C}_{1,2}^{\mathrm{SJ}(3\times12)}$ and the right-hand-side acceleration term $\boldsymbol{\xi}_{1,2}^{\mathrm{SJ}(3\times1)}$ are given as

$$\mathbf{C}_{1,2}^{\mathrm{SJ}(3\times12)} = \begin{bmatrix}\mathbf{I}_3 & -\mathbf{R}_1\tilde{\mathbf{r}}_{b_1}^{\mathrm{SJ}} & -\mathbf{I}_3 & \mathbf{R}_2\tilde{\mathbf{r}}_{b_2}^{\mathrm{SJ}}\end{bmatrix}, \tag{40}$$

and

$$\boldsymbol{\xi}_{1,2}^{\mathrm{SJ}(3\times1)} = \mathbf{R}_2\tilde{\boldsymbol{\omega}}_2\tilde{\boldsymbol{\omega}}_2\mathbf{r}_{b_2}^{\mathrm{SJ}} - \mathbf{R}_1\tilde{\boldsymbol{\omega}}_1\tilde{\boldsymbol{\omega}}_1\mathbf{r}_{b_1}^{\mathrm{SJ}}. \tag{41}$$

The generalized force vector $\mathbf{Q}$ comprises the external forces $\mathbf{Q}_{\mathrm{ext}}$ and nonlinear velocity terms $\mathbf{Q}_{\mathrm{nl}}$ that originate from the Euler equations according to the expression

$$\mathbf{Q} = \mathbf{Q}_{\mathrm{ext}} + \mathbf{Q}_{\mathrm{nl}}, \tag{42}$$

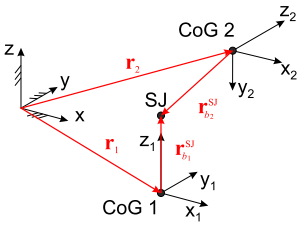


Fig. 17 Spherical joint position vectors

where the nonlinear velocity part is defined as

$$\mathbf{Q}_{\mathrm{nl}} = \left[\mathbf{0}_1^{(1\times 3)} \quad [-\tilde{\boldsymbol{\omega}}_1\mathbf{J}_1\boldsymbol{\omega}_1]^T \quad \mathbf{0}_2^{(1\times 3)} \quad [-\tilde{\boldsymbol{\omega}}_2\mathbf{J}_2\boldsymbol{\omega}_2]^T \quad \mathbf{0}_3^{(1\times 3)} \quad [-\tilde{\boldsymbol{\omega}}_3\mathbf{J}_3\boldsymbol{\omega}_3]^T \quad \mathbf{0}_4^{(1\times 3)} \quad [-\tilde{\boldsymbol{\omega}}_4\mathbf{J}_4\boldsymbol{\omega}_4]^T\right]^T, \tag{43}$$

and the part of external forces follows from all external forces that act on the system

$$\mathbf{Q}_{\mathrm{ext}} = \left[\mathbf{0}_1^{(1\times 3)} \quad -\mathbf{L}_{\mathrm{SJ}}^T \quad \mathbf{0}_2^{(1\times 3)} \quad [\mathbf{L}_{\mathrm{SJ}} - \mathbf{L}_{\mathrm{RJ}}]^T \quad \mathbf{0}_3^{(1\times 3)} \quad \mathbf{L}_{\mathrm{RJ}}^T \quad [\mathbf{R}_4\mathbf{F}_{\mathrm{PJ}}]^T \quad \mathbf{0}_4^{(1\times 3)}\right]^T. \tag{44}$$

In Eq. (44), the torque actuator $\mathbf{L}_{\mathrm{SJ}} = [-1 \ -3 \ -0.1]^T$ (acting along all three local axes of the joint) is imposed on the manipulator base spherical joint, whereas the revolute joint actuator is set as $\mathbf{L}_{\mathrm{RJ}} = [0\ 0.45\ 0]^T$. The force acting on the slider is collinear with the slider translation axis defined as $\mathbf{F}_{\mathrm{PJ}} = [0\ 0\ -0.533]^T$. The gravity is neglected.

In standard units, the mass and inertia tensor of the satellite body are set as $m_1 = 1800$ and $\mathbf{J}_1 = \mathrm{diag}[1800, 1800, 900]$. The base rod's mass and inertia tensor yield $m_2 = 80$ and $\mathbf{J}_2 = \mathrm{diag}[15.1125, 15.1125, 0.225]$, the slider rod's mass and inertia tensor are set as $m_3 = 60$ and $\mathbf{J}_3 = \mathrm{diag}[11.3, 11.3, 0.1]$, and the slider's mass and inertia tensor yield $m_4 = 30$ and $\mathbf{J}_4 = \mathrm{diag}[0.35, 0.35, 0.8937]$, respectively. The rheonomic constraint, which prescribes the rotational motion of the satellite's body, is specified with the global constant angular velocity set as $\boldsymbol{\omega}_{g1} = [\pi/60\ 0\ \pi/30]^T$. The initial conditions in the global coordinate system (see Fig. 16) are given as $\mathbf{r}_1^0 = [0\ 0\ 0]^T$, $\mathbf{r}_2^0 = [0\ 0\ 2.25]^T$, $\mathbf{r}_3^0 = [0\ 0\ 3.75]^T$, $\mathbf{r}_4^0 = \mathbf{r}_3^0$ and $\mathbf{R}_1^0 = \mathbf{R}_2^0 = \mathbf{R}_3^0 = \mathbf{R}_4^0 = \mathbf{I}$.

Similarly as it was done in the previous example, the proposed Lie-group integration procedure was repeated with different integration steps and stabilization parameters. First, a relative constraint violation error (31) was calculated for the constraints imposed on the system by the spherical joint (Fig. 17) only. This was done to allow for a direct comparison with the constraint violation stabilization results obtained within the heavy top example, which also only comprises a spherical joint.

To calculate the generalized position violation error for the satellite's spherical joint, the variables in (31) are defined as $\boldsymbol{\theta}^{\mathrm{int}} = \mathbf{r}_2$ and $\boldsymbol{\theta}^{\mathrm{con}} = \mathbf{R}_1\mathbf{r}_{b_1}^{\mathrm{SJ}} - \mathbf{R}_2\mathbf{r}_{b_2}^{\mathrm{SJ}}$ (based on the constraint equation (36) by taking into account $\mathbf{r}_1 = \mathbf{0}$), and the results for the integration step $h = 1\mathrm{e}{-5}$ and $Tol = 1\mathrm{e}{-14}$ are shown in Fig. 18. The same integration and stabilization parameters were also used in the heavy top example, where variables of (31) were defined on the basis of (25). The results are presented in Fig. 19.

When we compare the results in Figs. 18 and 19, it is visible that the stabilization procedure returned a less-profiled stabilization effect in the satellite example. Although the specified stabilization tolerance was always achieved, the final (overall) stabilization accuracy (obtained for a certain integration step after the specified accuracy had been achieved) was higher in the heavy top example than in the more complex satellite MBS (1e−24 vs.

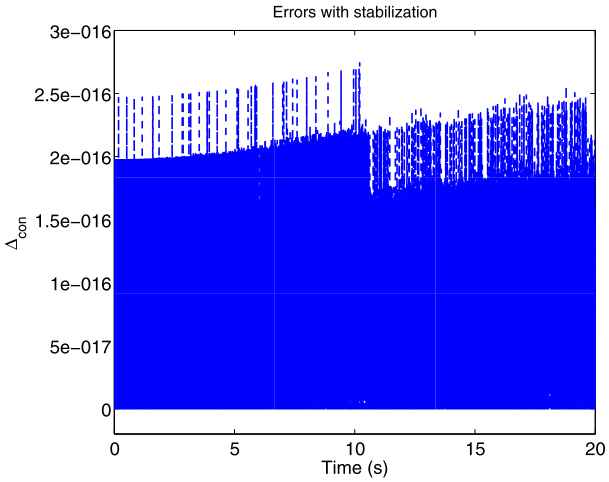


Fig. 18 Satellite spherical joint-constraint violation at configuration level, $h = 1e{-}5$, $Tol = 1e{-}14$

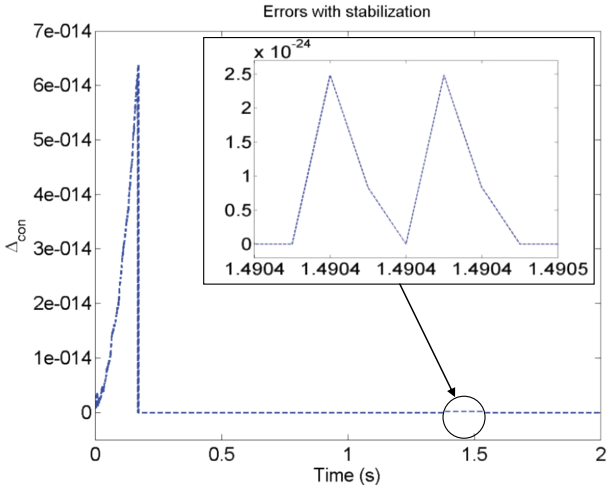


Fig. 19 Heavy top spherical joint-constraint violation at configuration level, $h = 1e{-}5$, $Tol = 1e{-}14$

1e−16). This can be explained by the kinematical complexity of the satellite MBS compared to the heavy top one-body mechanical system, which results in a higher number of integration variables and constraints, as well as in a larger dimension of the constrained optimization problem.

To calculate the violation error for the satellite spherical joint at the velocity level, the variables of (31) were defined as $\boldsymbol{\theta}^{\text{int}} = -\mathbf{v}_2$ and $\boldsymbol{\theta}^{con} = \mathbf{R}_1\tilde{\mathbf{r}}^{\text{SJ}}_{b_1}\boldsymbol{\omega}_1 - \mathbf{R}_2\tilde{\mathbf{r}}^{\text{SJ}}_{b_2}\boldsymbol{\omega}_2$ (based on Eq. (37) by taking into account $\mathbf{v}_1 = \mathbf{0}$), and stabilization is performed for the integration step $h = 1e{-}5$ with optimization tolerance set as $Tol = 1e{-}14$ and weighted matrix $\mathbf{W} = \mathbf{I}$. The results, depicted in Fig. 20, can be compared with the velocity constraint violation errors obtained with the same integration parameters in the heavy top example, presented in Fig. 21. Here, as it was the case regarding the generalized positions, the velocity projection algorithm has a smaller stabilization effect in the satellite example, compared to the heavy top velocity stabilization with the same integration and stabilization parameters.

Besides the relative constraint violation error for the spherical joint discussed above, the absolute constraint violation errors for the whole system were calculated via the equations

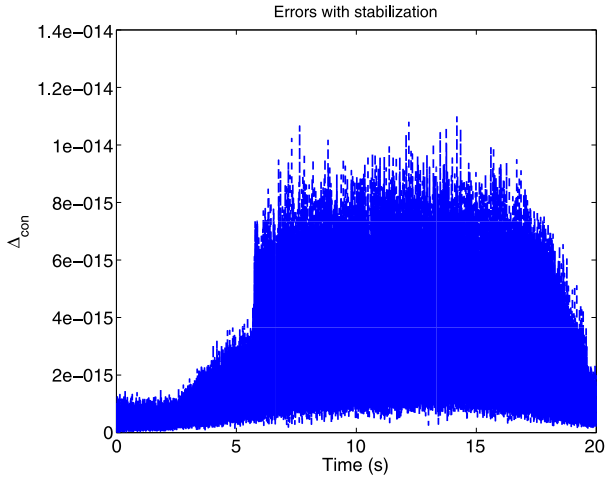


Fig. 20 Satellite spherical joint-constraint violation at velocity level, $h = 1e{-}5$, $Tol = 1e{-}14$

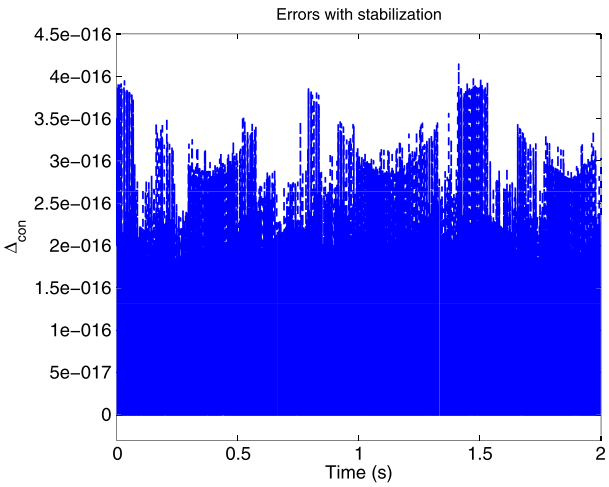


Fig. 21 Heavy top spherical joint-constraint violation at velocity level, $h = 1e{-}5$, $Tol = 1e{-}14$

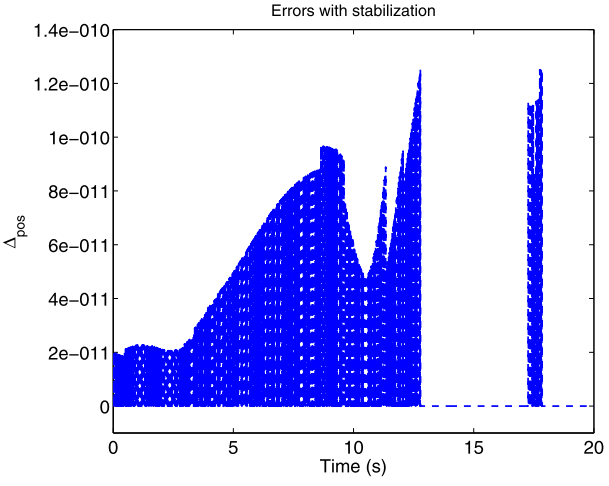


Fig. 22 Generalized position stabilization procedure, $h = 1e{-}2$, $Tol = 1e{-}10$

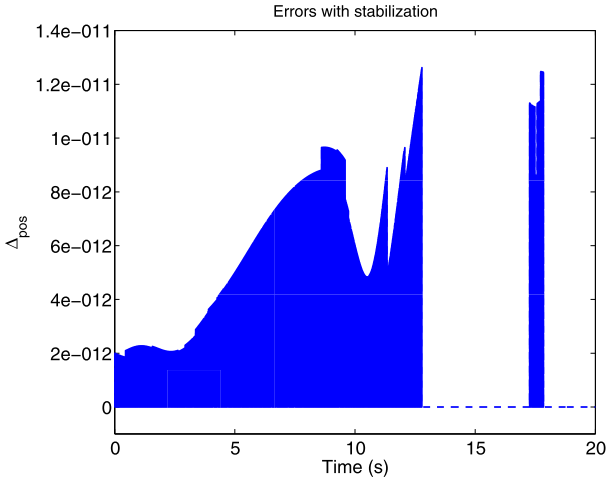


Fig. 23 Generalized position stabilization procedure, $h = 1e{-}3$, $Tol = 1e{-}11$

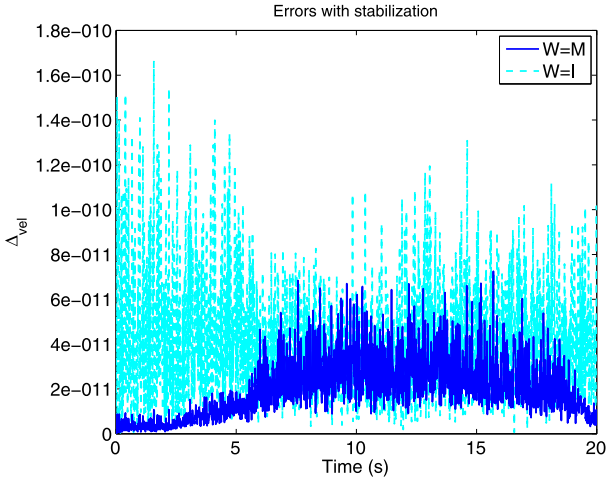


Fig. 24 Velocity stabilization procedure, $h = 1e{-}2$, $Tol = 1e{-}10$

$\mathbf{\Phi}(q) = \Delta_{\text{pos}}$ and $\dot{\mathbf{\Phi}}(q, \mathbf{v}) = \Delta_{\text{vel}}$ for the configuration and velocity constraints and depicted in Figs. 22, 23, 24 and 25. The integration time steps were $h = 1e{-}2$ and $h = 1e{-}3$, and optimization tolerances were set as $Tol = 1e{-}10$ and $Tol = 1e{-}11$, respectively, for both stabilization levels (note larger time steps and demanding tolerances). It is clear from these figures that the constraint violation error is controllable by the user via setting the *Tol* parameter at the desired value (in the context of generalized position stabilization and velocity stabilization with $\mathbf{W} = \mathbf{M}$ only some peaks exceeded—and only to a small extent—specified demanding tolerances within a three-step iteration process). It can also be seen in Figs. 24 and 25 that the velocity stabilization procedure, which used the norm weighted matrix equal to the system mass matrix $\mathbf{W} = \mathbf{M}$, yielded better results compared to those that were obtained with $\mathbf{W} = \mathbf{I}$. This effect is more pronounced here than in the heavy top example, and this might be explained by the fact that the satellite MBS comprises more bodies in the kinematical chain, meaning that the inertial characteristics of the system expressed in $\mathbf{M}$ influence the velocity constraint manifold projection given by (21) and (22) to a greater extent (especially in the context of integration with the larger time steps). However, this ex-

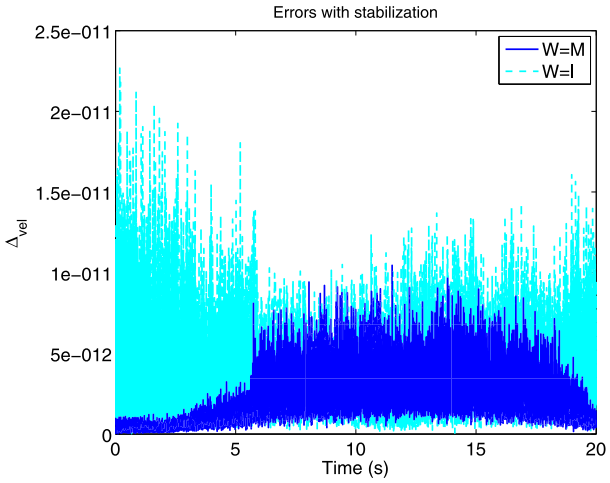


Fig. 25 Velocity stabilization procedure, $h = 1e{-}3$, $Tol = 1e{-}11$

planation should not be generalized since detailed performance of particular projections is clearly case-dependent (with the hint that the velocity projection with the weighted matrix $\mathbf{W} = \mathbf{M}$ generally returned better or equal results within the framework of this study).

It is important, however, that the desired stabilization effect could have been achieved, and this could have been done at minimal computational expense: although the tolerances imposed in the stabilization cases depicted in Figs. 22–25 were quite demanding ($Tol = 1e{-}10$, $Tol = 1e{-}11$) and the integration was performed with large integration steps, the presented results were obtained with numerical effort that never exceeded three iteration steps in the context of the iterative minimization procedure. This also means that if the constraint violation stabilization procedure is programmed separately ('in-house') within the framework of a general MBS code, the Newton iteration should be able to reuse the iteration Jacobian from the previous time steps, thus reducing the numerical effort of the stabilization.

8 Conclusion

A Lie-group integration method for a constrained MBS is proposed in the paper. The method operates on a system state space modelled as a Lie-group; the mathematical model of the MBS dynamics is modelled as DAE of index 1 on the Lie-group. Such an approach enables one to derive a coordinate-free mathematical formulation of the system dynamics, whereas the integration algorithm operates directly on the MBS state space manifold where the position and velocity constraints are introduced. Hence, no local parameters (such as Euler angles) for the mathematical description of large 3D rotations are required.

Since the proposed method operates directly with the MBS elements' angular velocities and rotational matrices, and local parameterization is avoided, this formulation circumvents the well-known problems of the classical vector-space-based MBS integration procedures, such as the kinematic singularity of the three-parameter rotation basis, re-parameterization of the system kinematics during integration, and numerical non-efficiency of the kinematic differential equations.

In order to eliminate a numerical constraint violation at the generalized positions and velocity level during the integration procedure (constraint violation stabilization is always

necessary if a mathematical model is shaped as a DAE system), the kinematical constraint violation stabilization algorithm on the introduced state space Lie-group is proposed. The stabilization algorithm is based on the constrained least-square minimization algorithm, and it stabilizes both configuration and velocity constraint violations.

The application of the proposed integration method is illustrated with two numerical examples. They showed that the procedure returns good and reliable simulation results. Since the method is formulated on the state space Lie-group, the $SO(3)$ matrices of the MBS elements' attitudes are obtained directly from the integrator, with their orthogonality properties preserved at the 'machine' (i.e. software) precision. Moreover, it was shown that the described constraint violation stabilization procedure allows for a controlled kinematical constraints violation, the tolerance of which can be specified by the user. Although iterative, the constraint stabilization procedure exhibited minimal computational burden in all analysed cases.

As a conclusion, the proposed Lie-group DAE-index-1 integration method is based on an efficient mathematical formulation. It is easy to use for discrete mechanical systems with kinematical constraints of general type, and it is especially recommended for an MBS with a large 3D rotation motion domain. Moreover, in the context of practical computations, an MBS dynamical model and imposed kinematical constraints are formulated in the matrix notation form that does not differ from the classical vector-space-based MBS formulations.

The proposed algorithm allows for synthesis of integration method of higher order of accuracy (4th order was used in the examples presented in the paper), based on the order of standard vector-space ODE integrator that is used for integration in a Lie-algebra. As an alternative to the described Lie-group state space MBS formulation, a state space formulation based on the $SE(3)$ Lie-group modelling of the MBS elements' screw motion was described in [38]. Although this approach has a clear advantage when system joints impose the bodies' relative motions that belong to a sub-group of $SE(3)$ (in these cases the kinematical constraints are numerically completely satisfied with the sole Lie-group integration, without applying a constraint violation stabilization method of any kind), for general MBS integration, both approaches have their pros and cons [38].

Acknowledgements The first and third author acknowledge the support of the Croatian Science Foundation under the contract of the project 04/35 'Geometric Numerical Integrators for Dynamic Analysis and Simulation of Structural Systems'. The second author acknowledges that this work has been partially supported by the Austrian COMET-K2 programm of the Linz Center of Mechatronics (LCM).

Appendix A: Differential of the exponential map

Starting from the rotational motion of one body, we introduce the differential of the exponential mapping $\mathrm{dexpm} : so(3) \times so(3) \to so(3)$ via 'left trivialized' tangent of the matrix exponential map 'expm' in a way that the following expression is valid:

$$\frac{\mathrm{d}}{\mathrm{d}t}\exp\mathrm{m}\big(\bar{w}(t)\big) = \exp\mathrm{m}\big(\bar{w}(t)\big)\mathrm{dexpm}_{-\bar{w}(t)}\big(\dot{\bar{w}}(t)\big), \tag{A.1}$$

where the function $\mathrm{dexpm}_{-\bar{w}}$ is defined as

$$\begin{aligned}\mathrm{dexpm}_{-\bar{w}}(w) &= w - \frac{1}{2!}[\bar{w}, w] + \frac{1}{3!}\big[\bar{w}, [\bar{w}, w]\big] + \frac{1}{4!}\big[\bar{w}, \big[\bar{w}, [\bar{w}, w]\big]\big] + \cdots \\ &= \sum_{j=0}^{\infty} \frac{1}{(j+1)!}\big(-\mathrm{ad}_{\bar{w}}^{j}(w)\big),\end{aligned} \tag{A.2}$$

and the adjoint operator $\mathrm{ad}_{\bar{w}}$ is given as the Lie bracket

$$\mathrm{ad}_{\bar{w}}(w) = \bar{w}w - w\bar{w} = [\bar{w}, w], \quad \text{for all } w(t), \bar{w}(t) \in so(3). \tag{A.3}$$

Furthermore, the inverse function $\mathrm{dexpm}^{-1}_{-\bar{w}}$ is defined by

$$\mathrm{dexpm}^{-1}_{-\bar{w}}(w) = w + \frac{1}{2}[\bar{w}, w] + \frac{1}{12}\big[\bar{w}, [\bar{w}, w]\big] + \cdots = \sum_{j=0}^{\infty} \frac{B_j}{j!}\big(-\mathrm{ad}^j_{\bar{w}}(w)\big), \tag{A.4}$$

where B_j are Bernoulli numbers [39]. Before we proceed, please note that Eqs. (A.2) and (A.4) are derived under the assumption of the 'left trivialization' expression in (A.1). This is in accordance with our formulation of the Lie-algebra $\mathcal{g}$, where the left-invariant vector field $\tilde{\boldsymbol{\omega}}_i \in so(3)$ is used in its definition. However, in the literature, the differential of the exponential mapping is usually defined by using the 'right trivialized' formulation in the form

$$\frac{\mathrm{d}}{\mathrm{d}t}\,\mathrm{expm}\big(\bar{w}(t)\big) = \mathrm{dexpm}_{\bar{w}(t)}\big(\dot{\bar{w}}(t)\big)\,\mathrm{expm}\big(\bar{w}(t)\big). \tag{A.5}$$

This would lead to expressions of $\mathrm{dexpm}_{\bar{w}}$ and $\mathrm{dexpm}^{-1}_{\bar{w}}$ with Lie brackets that would appear with different signs in (A.2) and (A.4). However, please note that if the right trivialization had been used, the final expressions for $\mathrm{dexpm}_{\bar{w}}$ and $\mathrm{dexpm}^{-1}_{\bar{w}}$ would have differed from (A.2) and (A.4) only in the sign of the second term $\pm\frac{1}{2}[\bar{w}, w]$.

After deriving the differential of the exponential map for the rotational motion of the rigid body, we need to include its translational part as well. Accordingly, the differential of the exponential mapping for the unconstrained body i motion $\mathcal{R}^3 \times SO(3)$ is given by

$$\mathrm{dexp}_{-\bar{z}_i}(z_i) = \mathrm{dexp}_{(-\bar{\mathbf{v}}_i, -\bar{w}_i)}(\mathbf{v}_i, w_i) = \big(\mathbf{v}_i, \mathrm{dexpm}_{-\bar{w}_i}(w_i)\big), \tag{A.6}$$

(where $\mathrm{dexpm}_{-\bar{w}_i}$ function is introduced as above), and its inverse is readily given as

$$\mathrm{dexp}^{-1}_{-\bar{z}_i}(z_i) = \mathrm{dexp}^{-1}_{(-\bar{\mathbf{v}}_i, -\bar{w}_i)}(\mathbf{v}_i, w_i) = \big(\mathbf{v}_i, \mathrm{dexpm}^{-1}_{-\bar{w}_i}(w_i)\big). \tag{A.7}$$

With the above results, we can define the differential of the exponential mapping for the whole unconstrained MBS system as the function $\mathrm{dexp} : s \times s \to s$ given by

$$\mathrm{dexp}_{-\bar{z}}(z) = \big(\mathrm{dexp}_{-\bar{z}_1}(z_1), \ldots, \mathrm{dexp}_{-\bar{z}_k}(z_k), \dot{\mathbf{v}}_1, \dot{w}_1, \ldots, \dot{\mathbf{v}}_k, \dot{w}_k\big), \tag{A.8}$$

and its inverse can be written as

$$\mathrm{dexp}^{-1}_{-\bar{z}}(z) = \big(\mathrm{dexp}^{-1}_{-\bar{z}_1}(z_1), \ldots, \mathrm{dexp}^{-1}_{-\bar{z}_k}(z_k), \dot{\mathbf{v}}_1, \dot{w}_1, \ldots, \dot{\mathbf{v}}_k, \dot{w}_k\big). \tag{A.9}$$

Appendix B: Derivation of algebraic equations imposed by the fixed-point constraint

In order to model fixed-point constraint, three algebraic constraint equations at the generalized displacement level can be written as

$$\boldsymbol{\Phi}^{\mathrm{FP}} = \mathbf{r}_1^{\mathrm{FP}} - \mathbf{r}_1 - \mathbf{R}_i \mathrm{r}_{b_1}^{\mathrm{FP}} = 0, \tag{B.1}$$

where $\mathbf{r}_{b_1}^{\mathrm{FP}}$ represents the location of the fixed-point constraint in the body 1 local coordinate system that is fixed to the body (see Fig. 26), $\mathbf{r}_1$ is the position of the mass centre of body

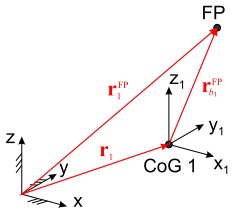


Fig. 26 Fixed-point constraint position vectors

1 in the global coordinate system, and $\mathbf{r}_1^{\mathrm{FP}}$ is a constant vector that determines, in the global coordinate system, the position of the body point that is fixed in space.

After differentiating with respect to time, the velocity level constraint is obtained in the form

$$\dot{\mathbf{\Phi}}^{\mathrm{FP}} = -\mathbf{v}_1 - \mathbf{R}_1\tilde{\boldsymbol{\omega}}_1\mathbf{r}_{b_1}^{\mathrm{FP}} = \mathbf{0}, \tag{B.2}$$

whereas additional differentiation yields an acceleration constraint in the form

$$\ddot{\mathbf{\Phi}}^{\mathrm{FP}} = -\dot{\mathbf{v}}_1 - \mathbf{R}_1\tilde{\mathbf{r}}_{b_1}^{\mathrm{FP}}\dot{\boldsymbol{\omega}}_1 - \mathbf{R}_1\tilde{\boldsymbol{\omega}}_1\tilde{\boldsymbol{\omega}}_1\mathbf{r}_{b_1}^{\mathrm{FP}} = \mathbf{0}. \tag{B.3}$$

To obtain the acceleration constraint in the form that can be included in the system dynamics governing equations (13) the acceleration constraint (B.3) can be written as

$$\mathbf{C}_1^{\mathrm{FP}(3\times6)}\dot{\mathbf{v}}_1^{(6\times1)} = \boldsymbol{\xi}_1^{\mathrm{FP}(3\times1)}, \tag{B.4}$$

where $\dot{\mathbf{v}}_1^{(6\times1)} = [\dot{\mathbf{v}}_1^T\ \dot{\boldsymbol{\omega}}_1^T]^T$ is the generalized acceleration vector of body 1, and the fixed-point constraint matrix $\mathbf{C}_1^{\mathrm{FP}(3\times6)}$ and the right-hand-side acceleration term $\boldsymbol{\xi}_1^{\mathrm{FP}(3\times1)}$ are given as

$$\mathbf{C}_1^{\mathrm{FP}(3\times6)} = \left[-\mathbf{I}_3 \quad \mathbf{R}_1\tilde{\mathbf{r}}_{b_1}^{\mathrm{FP}}\right], \tag{B.5}$$

and

$$\boldsymbol{\xi}_1^{\mathrm{FP}(3\times1)} = \mathbf{R}_1\tilde{\boldsymbol{\omega}}_1\tilde{\boldsymbol{\omega}}_1\mathbf{r}_{b_1}^{\mathrm{FP}}. \tag{B.6}$$

References

1. Schiehlen, W.: Multibody system dynamics: Roots and perspectives. Multibody Syst. Dyn. **1**, 149–188 (1997)
2. Marsden, J.E., Ratiu, T.S.: Introduction to Mechanics and Symmetry. Springer, Berlin (1999)
3. Holm, D.: Geometric Mechanics. Part II: Rotating, Translating and Rolling. Imperial College Press, London (2008)
4. Nikravesh, P.E.: Computer-Aided Analysis of Mechanical Systems. Prentice Hall, Englewood Hills (1988)
5. Shutz, B.F.: Geometrical Methods of Mathematical Physics. Cambridge Univ. Press, Cambridge (1980)
6. Boothby, W.: An Introduction to Differentiable Manifolds and Riemannian Geometry, 2nd edn. Academic Press, San Diego (2003)
7. Morawiec, A.: Orientations and Rotations. Springer, Berlin (2004)
8. Betsch, P., Siebert, R.: Rigid body dynamics in terms of quaternions: Hamiltonian formulation and conserving numerical integration. Int. J. Numer. Methods Eng. **79**, 444–473 (2009)
9. Reich, S., Zentrum, K.Z.: Symplectic integrators for systems of rigid bodies. Integration Algorithms for Classical Mechanics. Fields Inst. Commun. **10**, 181–191 (1996)
10. Leimkuhler, B., Reich, S.: Simulating Hamiltonian Dynamics. Cambridge Univ. Press, Cambridge (2004)

11. Betsch, P., Steinmann, P.: Constrained integration of rigid body dynamics. Comput. Methods Appl. Mech. Eng. **191**, 467–488 (2001)
12. Iserles, A., Munthe-Kaas, H.Z., Norsett, S.P., Zanna, A.: Lie-group methods. Acta Numer. **9**, 215–365 (2000)
13. Hairer, E., Lubich, C., Wanner, G.: Geometric Numerical Integration. Springer, Berlin (2006)
14. Munthe-Kaas, H.: Runge–Kutta methods on Lie groups. BIT Numer. Math. **38**, 92–111 (1998)
15. Brüls, O., Cardona, A.: On the use of Lie group time integrators in multibody dynamics. ASME J. Comput. Nonlinear Dyn. **5**, 1–23 (2010)
16. Cardona, A., Geradin, M.: A beam finite element non-linear theory with finite rotations. Int. J. Numer. Methods Eng. **26**, 2403–2438 (1988)
17. Simo, J.C., Vu-Quoc, L.: On the dynamics in space of rods undergoing large motions—geometrically exact approach. Comput. Methods Appl. Mech. Eng. **66**, 125–161 (1988)
18. Simo, J., Wong, K.: Unconditionally stable algorithms for rigid body dynamics that exactly preserve energy and momentum. Int. J. Numer. Methods Eng. **31**, 19–52 (1991)
19. Bottasso, C.L., Borri, M.: Integrating finite rotations. Comput. Methods Appl. Mech. Eng. **164**, 307–331 (1998)
20. Müller, A.: Approximation of finite rigid body motions from velocity fields. J. Appl. Math. Mech./Z. Angew. Math. Mech. (ZAMM) **90**, 514–521 (2010)
21. Müller, A., Terze, Z.: The significance of the configuration space Lie group for the constraint satisfaction in numerical time integration of multibody systems. Mech. Mach. Theory **82**, 173–202 (2014)
22. Haug, E.J.: Computer Aided Kinematics and Dynamics of Mechanical Systems. Allyn and Bacon, Boston (1989)
23. Schiehlen, W.: Multibody Systems Handbook. Springer, Berlin (1990)
24. Blajer, W.: A geometrical interpretation and uniform matrix formulation of multibody system dynamics. Z. Angew. Math. Mech. **81**(4), 247–259 (2001)
25. Terze, Z., Naudet, J.: Geometric properties of projective constraint violation stabilization method for generally constrained multibody systems on manifolds. Multibody Syst. Dyn. **20**, 85–106 (2008)
26. Geradin, M., Cardona, A.: Flexible Multibody Dynamics. Wiley, Chichester (2004)
27. Bauchau, O.A.: Flexible Multibody Dynamics. Springer, Dordrecht, Heidelberg, London, New York (2010)
28. Celledoni, E., Owren, B.: Lie group methods for rigid body dynamics and time integration on manifolds. Comput. Methods Appl. Mech. Eng. **192**, 421–438 (2003)
29. Blajer, W.: Methods for constraint violation suppression in the numerical simulation of constrained multibody systems—a comparative study. Comput. Methods Appl. Mech. Eng. **200**(13–16), 1568–1576 (2011)
30. García Orden, J.C.: Energy considerations for the stabilization of constrained mechanical systems with velocity projection. Nonlinear Dyn. **60**(1–2), 49–62 (2010)
31. Eich, E.: Convergence results for a coordinate projection method applied to mechanical systems with algebraic constraints. SIAM J. Numer. Anal. **30**, 1467–1482 (1993)
32. Eich-Soellner, E., Führer, C.: Numerical Methods in Multibody Dynamics. Teubner, Stuttgart (1998)
33. Bayo, E., Ledesma, R.: Augmented Lagrangian and mass orthogonal projection methods for constrained multibody dynamics. Nonlinear Dyn. **9**, 113–130 (1996)
34. Terze, Z., Naudet, J.: Structure of optimized generalized coordinates partitioned vectors for holonomic and non-holonomic systems. Multibody Syst. Dyn. **24**, 203–218 (2010)
35. Hairer, E., Wanner, G.: Solving Ordinary Differential Equations. II. Stiff and Differential-Algebraic Problems. Springer, Berlin (1996)
36. Müller, A., Terze, Z.: A constraint stabilization method for time integration of constrained multibody systems in Lie group setting. In: ASME 2014 IDETC on 10th International Conference on Multibody Systems, Nonlinear Dynamics, and Control (MSNDC), 17–20 August 2014, Buffalo, New York, USA (2014)
37. Andrzejewski, T., Bock, H.G., Eich, E., von Schwerin R.: Recent advances in the numerical integration of multibody systems. In: Schiehlen, W. (ed.) Advanced Multibody System Dynamics. Kluwer Academic, Dordrecht (1993)
38. Müller, A., Terze, Z.: On the choice of configuration space for numerical Lie group integration of constrained rigid body systems. J. Comput. Appl. Math. (2013). doi:10.1016/j.cam.2013.10.039
39. Budd, C.J., Iserles, A.: Geometric integration: numerical solution of differential equations on manifolds. Philos. Trans.: Math. Phys. Eng. Sci. **357**, 945–956 (1999)