DAE Index 1 Formulation for Multibody System Dynamics in Lie-Group Setting

Zdravko Terze*, Andreas Müller#, Dario Zlatar*

* Faculty of Mechanical Engineering and
Naval Architecture
University of Zagreb
I. Lučića 5, 10002 Zagreb, Croatia
zdravko.terze@fsb.hr,
dario.zlatar@fsb.hr

Institute of Mechatronics
Reichenhainer Str. 88,
09126 Chemnitz, Germany
andreas.mueller@ifm-chemnitz.de

ABSTRACT

A Lie-group integration method for constrained multibody systems is proposed in the paper and applied for numerical simulation of a satellite dynamics. Mathematical model of multibody system dynamics is shaped as DAE system of equations of index 1, while dynamics is evolving on Lie-group introduced as system 'state-space formulation'. The basis of the method is Munthe-Kaas algorithm for ODE on Lie-groups, which is re-formulated and expanded to be applicable for the integration of constrained multibody dynamics in DAE index 1 form. The constraint violation stabilization algorithm at the generalized position and velocity level is introduced by using two different algorithms: a first one that operates directly on the 'state-space' manifold and, a second one, that uses Cartesian rotation vectors as local coordinates for the generalized positions. A numerical example of 'dual-spin' satellite that demonstrates the proposed integration procedure is described and discussed at the end of the paper.

1. INTRODUCTION

Design of the numerical integration methods that operate on manifolds and Lie-groups, instead of linear vector spaces like 'standard' ODE and DAE integrators, offer some attractive features such as numerical robustness and avoidance of the kinematical singularities as well as numerical efficiency of the code. Along this line, the goal of the paper is to propose mathematical model and numerical integration algorithm based on Munthe-Kaas Lie-group integration scheme [10, 11] that will be pertinent to multibody dynamics application cases. The state space of multibody system is modeled as a manifold (Lie-group) and mathematical model of the numerical scheme is shaped as DAE of index 1. The method is primarily focused on dynamics of rigid body multibody systems, although its application can be easily extended also on the systems that possess elastic components.

2. GEOMETRIC DAE INTEGRATION PROCEDURE FOR MBS

2.1. Lie-group ODE integrator

The configuration of a rotating rigid body is given by a rotation matrix $\mathbf{R}$ that belongs to Special Orthogonal Group *SO*(3):

$$SO(3) = \left\{ \mathbf{R} \in \mathcal{R}^{3x3} : \mathbf{R}\mathbf{R}^T = \boldsymbol{I}, \det \mathbf{R} = +1 \right\}. \tag{1}$$

From the geometrical point of view, *SO*(3) can be considered as a differential manifold (space with different possibilities of parametarizations on which we can do calculus). Tangent vectors $\dot{\mathbf{R}}$ to *SO*(3) at point $\mathbf{R} \in SO(3)$ of the manifold belong to the tangent space $T_{\mathbf{R}}SO(3)$, $\dot{\mathbf{R}} \in T_{\mathbf{R}}SO(3)$. Moreover, *SO*(3) has the properties of Lie-group, where the tangent space at the group identity $\boldsymbol{I}$ has an additional structure. This vector space is equipped with matrix commutator and constitutes Lie-algebra of *SO*(3), the set of skew-symmetric matrices denoted by *so*(3). The element of Lie-algebra $\tilde{\boldsymbol{\omega}} \in so(3)$ can be

identified with $\mathcal{R}^3$ *via* mapping operator which maps a vector $\boldsymbol{\omega} \in \mathcal{R}^3$ to a matrix $\tilde{\boldsymbol{\omega}} \in so(3)$. Between the elements of the group $\mathbf{R} \in SO(3)$ and Lie-algebra $\tilde{\boldsymbol{\omega}} \in so(3)$ exists natural correspondence *via* exponential map [5, 4, 9] and the expression $\mathbf{R}(t) = \exp(t\tilde{\boldsymbol{\omega}})$ is solution of the inital value problem

$$\dot{\mathbf{R}}(t) = \mathbf{R}(t)\tilde{\boldsymbol{\omega}}, \ \mathbf{R}(0) = \boldsymbol{I}. \tag{2}$$

Also, the Lie-algebra element $\tilde{\boldsymbol{\omega}}$ defines left-invariant vector field $\mathbf{X}_{\tilde{\boldsymbol{\omega}}}$ on *SO*(3) *via* relation $\mathbf{X}_{\tilde{\boldsymbol{\omega}}}(\mathbf{R}) = L'_{\mathbf{R}}(\tilde{\boldsymbol{\omega}}), \mathbf{X}_{\tilde{\boldsymbol{\omega}}}(\mathbf{R}) \in T_{\mathbf{R}}SO(3)$, where the tangent map $L'_{\mathbf{R}}(\tilde{\boldsymbol{\omega}}) : TSO(3) \to TSO(3)$ is given as $L'_{\mathbf{R}}(\tilde{\boldsymbol{\omega}}) = \mathbf{R}\tilde{\boldsymbol{\omega}}$ defining left kinematic Poisson relation $\dot{\mathbf{R}} = \mathbf{R}\tilde{\boldsymbol{\omega}}$. Equivalently, instead of using body coordinates $\tilde{\boldsymbol{\omega}}$, the right Poisson equation can be formulated as $\dot{\mathbf{R}} = \tilde{\boldsymbol{\omega}}_S\mathbf{R}$ by using angular velocity expressed *via* spatial coordinates $\tilde{\boldsymbol{\omega}}_S$ [8]. As explained above, left and right Poisson kinematic equation represent differential equation on *SO*(3), since $\mathbf{R} \in SO(3)$ and the angular velocity tensor ($\tilde{\boldsymbol{\omega}}$ or $\tilde{\boldsymbol{\omega}}_S$) belongs to Lie-algebra *so*(3). In the case when angular velocity is not a constant skew-symmetric matrix (as it is the case in the equation (2), which has solution $\mathbf{R}(t) = \exp(t\tilde{\boldsymbol{\omega}})$), body kinematics that evolves on *SO*(3) is to be computed numerically. In the general setting, the objective is to find solution of differential equation on a matrix Lie-group *G*, with Lie-algebra ***g***, in the form

$$\dot{Y}(t) = A(Y(t))Y(t), \tag{3}$$

where $Y(0) \in G$ and $A(Y) \in g$ for all $Y \in G$ and right trivialization is used. The equation (3) can be numerically solved by using different geometric integration methods [5, 4], such as
Munthe-Kaas algorithm [4, 11] that assumes result in the form

$$Y(t) = \exp(u(t))Y_0, \tag{4}$$

where $u(t)$ is the solution of

$$\dot{u} = d\exp_u^{-1}(A(Y(t))), \ u(0) = 0. \tag{5}$$

By following this route, the numerical solution of kinematic equation (3) can be incorporated into the computational procedures based on Newton-Euler formulation, where rotational dynamics of rigid body is studied directly on *SO*(3). This leads to more efficient procedures since no local parametarisation of 3D rotation is needed.

2.2. Lie-group DAE integration procedure

2.2.1. *Procedure framework*

The configuration space of MBS is modeled as $G = \mathcal{R}^3 \times ... \times \mathcal{R}^3 \times SO(3) \times ... \times SO(3)$, which is a *n*-dimensional manifold with Lie-group properties, consisting of translational and rotational kinematical domains of each rigid body in MBS. The Lie-group composition operation $G \times G \to G$ is introduced by $p_{com} = p_1 \circ p_2$, where $p_1, p_2, p_{com} \in G$ and the identity element *e* of the group is defined as $p \circ e = e \circ p = p, \forall p \in G$. The Lie-algebra $\mathfrak{g} = T_eG$ (vector space that is isomorphic to $\mathcal{R}^n$) is defined as the tangent space T_pG at the identity *p=e*. The tangent vector in T_pG (at any point $p \in G$) can be represented in Lie-algebra $\mathfrak{g}$ *via* derivation L'_p of the left translation map $L_p : G \to G, y \mapsto p \circ y$. Thus, for *y=e*, we can define bijection $L'_p(e) : \mathfrak{g} \to T_pG, \tilde{\boldsymbol{\Omega}} \mapsto L'_p(e) \cdot \tilde{\boldsymbol{\Omega}}$, where $L'_p(e) \cdot \tilde{\boldsymbol{\Omega}}$ is

directional derivative of L_p at the point $y=e$ in direction of $\tilde{\boldsymbol{\Omega}} \in \mathfrak{g}$ (since G is Lie-group, the element of Lie-algebra $\tilde{\boldsymbol{\Omega}}$ defines left invariant vector field on G, similarly as it was the case with $SO(3)$).

To incorporate kinematical constraints of MBS, the function $\boldsymbol{\Phi}: G \to \mathcal{R}^m$ are imposed on G, meaning that system is constrained to evolve on the n-m dimensional sub-manifold $S = \{p \in G : \boldsymbol{\Phi}(p) = 0\}$. Consequently, dynamic equations of MBS are shaped in the form [1]

$$\begin{aligned} \mathbf{M}(p)\dot{\mathbf{v}} + \mathbf{Q}(p, \mathbf{v}, t) + \mathbf{C}^T(p)\boldsymbol{\lambda} = 0 \\ \boldsymbol{\Phi}(p) = 0 \\ \dot{p} = L'_p(e) \cdot \tilde{\mathbf{v}}, \end{aligned} \tag{6}$$

where $\mathbf{M}$ is $n \times n$ dimensional inertia matrix, $\mathbf{v} \in \mathcal{R}^n$, $\mathbf{v} = [\mathbf{v}_1, ..., \mathbf{v}_k, \boldsymbol{\omega}_1, ..., \boldsymbol{\omega}_k]^T$ are system velocities (k bodies are assumed), $\mathbf{Q}$ represents external and non-linear velocity forces, $\boldsymbol{\lambda} \in \mathcal{R}^m$ is the vector of Lagrangian multipliers and $\mathbf{C}$ is $m \times n$ dimensional constraint gradient matrix, such that $\boldsymbol{\Phi}'(p) \cdot \tilde{\boldsymbol{\Omega}} = \mathbf{C}(p)\boldsymbol{\Omega}, \forall \boldsymbol{\Omega} \in \mathcal{R}^n$ is valid. The equation (6) represents DAE system of index 3. Within the framework of the proposed integration procedure, the equation (6) will be re-shaped into the DAE of index 1 form by including kinematical constraints at the acceleration level $\ddot{\boldsymbol{\Phi}}(p, \mathbf{v}, \dot{\mathbf{v}}) = 0$ (instead of $\boldsymbol{\Phi}(p) = 0$) and integrated by the routine based on the Munthe-Kaas algorithm.

To ensure that kinematical constraints are satisfied during integration, constraint violation of the system velocities $\mathbf{v} \in \mathcal{R}^n$ and generalized positions $p \in G$ will be corrected by using stabilization algorithm.

2.2.2. Integration algorithm

As introduced, the configuration space of MBS is modeled as Lie-group $G = \mathcal{R}^3 \times ... \times \mathcal{R}^3 \times SO(3) \times ... \times SO(3)$ where translation and rotation of each rigid body is included in the n-dimensional configuration domain and element of the group is given as $p = (\mathbf{x}_1, ..., \mathbf{x}_k, \mathbf{R}_1, ..., \mathbf{R}_k)$. However, since proposed time integration routine operates 'simultaneously' at the generalized positions and generalized velocities, a system has to be modeled on the $2n$-dimensional Lie-group $S = \mathcal{R}^3 \times ... \times \mathcal{R}^3 \times SO(3) \times ... \times SO(3) \times ... \times \mathcal{R}^3 \times ... \times \mathcal{R}^3 \times so(3) \times ... \times so(3) \cong TG$
(a system 'state space') with the element

$$q = (\mathbf{x}_1, ..., \mathbf{x}_k, \mathbf{R}_1, ..., \mathbf{R}_k, \mathbf{v}_1, ..., \mathbf{v}_k, \tilde{\boldsymbol{\omega}}_1, ..., \tilde{\boldsymbol{\omega}}_k), \tag{7}$$

and its Lie-algebra $\mathcal{S} = \mathcal{R}^3 \times ... \times \mathcal{R}^3 \times so(3) \times ... \times so(3) \times ... \times \mathcal{R}^3 \times ... \times \mathcal{R}^3 \times \mathcal{R}^3 \times ... \times \mathcal{R}^3$ with the element given as

$$z = (\mathbf{v}_1, ..., \mathbf{v}_k, \tilde{\boldsymbol{\omega}}_1, ..., \tilde{\boldsymbol{\omega}}_k, \dot{\mathbf{v}}_1, ..., \dot{\mathbf{v}}_k, \dot{\tilde{\boldsymbol{\omega}}}_1, ..., \dot{\tilde{\boldsymbol{\omega}}}_k). \tag{8}$$

By confining ourselves on a single body system to keep formulation short, we introduce operations in Lie-group S and its Lie-algebra $\mathcal{S}$ as follows.

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

Addition in $\mathcal{S}$: $(v, w, c, d) + (\bar{v}, \bar{w}, \bar{c}, \bar{d}) = (v + \bar{v}, w + \bar{w}, c + \bar{c}, d + \bar{d})$.

Multiplication by scalar in $\mathcal{S}$: $\alpha(v, w, c, d) = (\alpha v, \alpha w, \alpha c, \alpha d)$.

Exponential map in $\mathcal{S}$: $\exp(v, w, c, d) = (v, \exp(w), c, d)$.

Bracket in $\mathcal{S}$: $[(v, w, c, d), (\bar{v}, \bar{w}, \bar{c}, \bar{d})] = (0, w \times \bar{w}, 0, 0)$.

Here, on the right hand side of definitions, '$\cdot$' is the multiplication in *SO*(3), '+' is addition in $\mathcal{R}^3$ and *so*(3) and exp is exponential map on *so*(3). The operations in $\mathcal{S}$ and S of multibody system, consisting of system of *k* bodies, is defined component-wise equivalently as for a single body system.

With all Lie-group operations in place, a differential equation describing dynamics of MBS on Lie group S can be written in the form

$$\dot{q} = F(q)q\,, \tag{9}$$

where $q \in S$ and $F: S \to \mathcal{S}$ is given by $F: q \to z$, where elements *q* and *z* are given by (7) and (8).

During evaluation of $F: q \to z$, the variables $\dot{\mathbf{v}} = \left[\dot{\mathbf{v}}_1,...,\dot{\mathbf{v}}_k, \dot{\tilde{\boldsymbol{\omega}}}_1,...,\dot{\tilde{\boldsymbol{\omega}}}_k\right]^T$ are determined by the system dynamics equation

$$\begin{bmatrix} \mathbf{M} & \mathbf{C}^T \\ \mathbf{C} & \mathbf{0} \end{bmatrix}\begin{bmatrix} \dot{\mathbf{v}} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{Q} \\ \boldsymbol{\xi} \end{bmatrix}, \tag{10}$$

which has to be solved (linear algebraic system for variables $\dot{\mathbf{v}}$ and $\boldsymbol{\lambda}$) within integration algorithm of the differential equation (9).

The equation (10) represents first two equations of system (6), shaped as DAE of index 1, where the acceleration kinematical constraints $\ddot{\boldsymbol{\Phi}}(p, \mathbf{v}, \dot{\mathbf{v}}) = 0$ are introduced as $\mathbf{C}\dot{\mathbf{v}} = \boldsymbol{\xi}$ [12].

The differential equation (9) of the system dynamics on S has the same form as differential equation (3) and can be solved by using Munthe-Kaas (MK) type of integration algorithm [10, 11]. Similarly as it was the case with (3), (9) can be solved by introducing local integration coordinate *u* in $\mathcal{S}$ that satisfy

$$\dot{u} = d\exp_u^{-1}(F(q))\,,\ u(0) = 0\,. \tag{11}$$

Within MK method, (9) is integrated in $\mathcal{S}$ and numerical solution is than reconstructed on S via exponential mapping. The algorithm itself can be given in the form [11]

$q_0 = q_{w-1}$
for $i = 1, 2, \ldots, s$
  $u_i = h\sum_{j=1}^{i-1} a_{ij}\tilde{k}_j$
  $k_i = F(c_i h, \exp(u_i, q_0))$
  $\tilde{k}_i = \mathrm{dexpinv}(u_i, k_i, n)$
end
$v = h\sum_{j=1}^{s} b_j\tilde{k}_j$
$q_w = \exp(v, q_0)$

where the coefficients are given by the classical *s*-stage *n*th order Runge-Kutta method's Butcher table [10, 11, 7] and function dexpinv is defined as follows [5, 4, 11]

$$\mathrm{dexpinv}(u, k, n) = k - \frac{1}{2}[u, k] + \sum_{p=2}^{n-1}\frac{B_p}{p!}\overbrace{[u,[u,[\ldots,}^{p}[u, k]]]].$$

The variables $u_i, k_i, \tilde{k}_i$ are MK method internal integration variables [11, 3] (similar as those that are defined within the framework of RK algorithms - MK methods reduce to RK algorithms when operate in vector space), which have the same format as *z* given by (8), see also (11).

2.3. Constraint violation stabilization procedure

A numerical solution obtained by the described algorithm will satisfy constraint equation at the acceleration level $\ddot{\mathbf{\Phi}}(p,\mathbf{v},\dot{\mathbf{v}})=0$ automatically, since this equation is directly incorporated in the function evaluation $F: S \to \mathcal{S}$ *via* formulation (10). However, the constraint equations for the generalized positions $\mathbf{\Phi}(p)=0$ and velocities $\dot{\mathbf{\Phi}}(p,\mathbf{v})=0$ (which can be also written in the form $\mathbf{C}(p)\mathbf{v}=\xi$ [12]) that are also part of DAE formulation (6), will be unavoidably violated during the straightforward integration based on the MK type of algorithm.

For the purpose of constraint stabilization procedure that operates directly on S, we propose a projective method that is based on nonlinear constrained least square problem given in the form

$$\min_{(p_k,\mathbf{v}_k)} \left\| \begin{pmatrix} p_k - \hat{p}_k \\ \mathbf{v}_k - \hat{\mathbf{v}}_k \end{pmatrix} \right\|_{\mathbf{w}}^2, \ \mathbf{\Phi}(p)=0, \ \mathbf{R}_i\mathbf{R}_i^{\,T}=\boldsymbol{I}, \ \dot{\mathbf{\Phi}}(p,\mathbf{v})=0, \tag{12}$$

(where $\| \ \|_{\mathbf{w}}$ denotes the weighted norm). The projected variables have to satisfy constraint equations $\mathbf{\Phi}(p)=0$, $\mathbf{R}_i\mathbf{R}_i^{\,T}=\boldsymbol{I}$ and $\dot{\mathbf{\Phi}}(p,\mathbf{v})=0$ to obtain stabilized values p_k ,$\mathbf{v}_k$, as expressed in, while $\hat{p}_k, \hat{\mathbf{v}}_k$ are 'un-stabilized' values that are obtained from the integrator for the current integration step. It should be emphasized here that described constraint stabilization at the velocity level (that brings stabilized value $\mathbf{v}_k$ from the initial integration value $\hat{\mathbf{v}}_k$) is a 'linear' one-step projective process, while 'generalized position' p_k stabilization requires iterative procedure.

In the sequel, a linear procedure also for 'position' variables, based on Cartesian rotation vectors [2] that are related with $\mathbf{R}_i \in SO(3)$ by the equation

$$\mathbf{R}_i = 1 + \tilde{\mathbf{\Psi}}_i + \frac{1}{2}\tilde{\mathbf{\Psi}}_i^{\,2} + \ldots = \exp(\tilde{\mathbf{\Psi}}_i), \tag{13}$$

will be introduced. Along this line, to design non-iterative constraint stabilization procedure for the variables $p=(\mathbf{x}_1,\ldots,\mathbf{x}_k,\mathbf{R}_1,\ldots,\mathbf{R}_k)$, we note that stabilized constraint equation at the generalized position level should read

$$\mathbf{\Phi}(p)=0, \tag{14}$$

but, because of the numerical errors, current integration values of *p* do not satisfy (14) completely, yielding

$$\mathbf{\Phi}(\hat{p})\neq 0, \tag{15}$$

where $\hat{p}$ are 'un-stabilized' values that are results of the current integration step. To calculate 'final' (stabilized) values of *p* that satisfy (14) for the current step, we write.

$$p=(\hat{\mathbf{x}}_1+\Delta\mathbf{x}_1,\ldots,\hat{\mathbf{x}}_k+\Delta\mathbf{x}_k,\hat{\mathbf{R}}_1\exp(\Delta\tilde{\mathbf{\Theta}}_1),\ldots,\hat{\mathbf{R}}_k\exp(\Delta\tilde{\mathbf{\Theta}}_i)), \tag{16}$$

where $\Delta\mathbf{x}_i, \Delta\tilde{\mathbf{\Theta}}_i$ are correction values that have to be introduced to bring $\hat{p}$ in accordance with (14) . In (16), $\Delta\mathbf{x}_i$ is a 'standard' vector correction of the linear displacement (body position), while $\Delta\tilde{\mathbf{\Theta}}_i$ is Lie-algebra vector component representation (expressed in skew-symmetric matrix/tensor form pertinent to *so(3)* by using body-fixed coordinate system) of a small rotation correction vector $\Delta\mathbf{\Theta}_{\mathbf{R}_i}$ needed to adjust orientation of the body *i* to be in accordance with the (14). Furthermore, by following [2], the variations $\delta\mathbf{\Theta}_{\mathbf{R}_i}$ and $\delta\mathbf{\Psi}_i$ are related by

$$\delta\boldsymbol{\Theta}_{\mathbf{R}_i} = T(\boldsymbol{\Psi}_i)\delta\boldsymbol{\Psi}_i\,, \tag{17}$$

where $T(\boldsymbol{\Psi}_i)$ is a tangent operator that relates tangent spaces $T_I SO(3)$ and $T_{\mathbf{R}_i} SO(3)$, see [6, 2] for the details.

To proceed with the design of the non-iterative 'position' constraint stabilization procedure, we will adopt the algorithm proposed in [13] and adjust it to be valid within the manifold $\mathcal{S}$ computational framework. Since 'original' constraint equation (14) is not satisfied due to the integration numerical errors, it is proposed in [13] to expand (14) by adding variation correction term as

$$\hat{\boldsymbol{\Phi}} + \delta\boldsymbol{\Phi} = 0\,. \tag{18}$$

The equation (18) is then used as a basic equation for calculation of the necessary increments of the system coordinates by linearising $\delta\boldsymbol{\Phi}$ and keeping only first variation of the coordinates into consideration. However, in the case of calculation on $\mathcal{S}$ *via* variables *p*, we can not proceed with equation (18) directly since $\mathcal{S}$ is a manifold (a non-linear space) and addition that would combine evaluation of the function $\boldsymbol{\Phi}$ at the different points on the manifold is not a defined operation.

Therefore, we will adopt vectorial representation of the rotations by using $\boldsymbol{\Psi}_i$ instead of $\mathbf{R}_i \in SO(3)$ and, by noticing that we can write $\boldsymbol{\Phi}(\hat{p}) = \boldsymbol{\Phi}(\hat{\boldsymbol{q}})$, the equation (18) can be written in the 'local' vectorial form

$$\boldsymbol{\Phi}(\hat{p}) + (\partial\boldsymbol{\Phi}/\partial\boldsymbol{q})\begin{bmatrix}\delta\mathbf{x}\\ \delta\boldsymbol{\Psi}\end{bmatrix} = 0\,, \tag{19}$$

where we adopted local system coordinates as $\boldsymbol{q} = [\mathbf{x} \quad \boldsymbol{\Psi}]$. By solving equation (19) for the variations $\delta\mathbf{x}, \delta\boldsymbol{\Psi}$, after assuming that correction increments are small enough that can be substituted by the first variations i.e. $\Delta\mathbf{x} \approx \delta\mathbf{x}, \Delta\boldsymbol{\Psi} \approx \delta\boldsymbol{\Psi}$, we are able to find correction terms that *via* relation $\boldsymbol{q} = \hat{\boldsymbol{q}} + \delta\boldsymbol{q}$ i.e.

$$\begin{bmatrix}\mathbf{x}\\ \boldsymbol{\Psi}\end{bmatrix} = \begin{bmatrix}\hat{\mathbf{x}}\\ \hat{\boldsymbol{\Psi}}\end{bmatrix} + \begin{bmatrix}\Delta\mathbf{x}\\ \Delta\boldsymbol{\Psi}\end{bmatrix}, \tag{20}$$

bring unstabilized values $\hat{\boldsymbol{q}}$ into the stabilized variable $\boldsymbol{q}$ that satisfy 'position' constraint equation $\boldsymbol{\Phi}(\boldsymbol{q}) = 0$ for the current integration step.

However, the equation (19) is undetermined and, to make it solvable, we will assume that correction terms $\delta\mathbf{x}, \delta\boldsymbol{\Psi}$ are completelly 'sunk' into the constrained subspace that are spanned by the rows of the constraint matrix $(\partial\boldsymbol{\Phi}/\partial\boldsymbol{q})^T$. Thus, we can write

$$\delta\boldsymbol{q} = (\partial\boldsymbol{\Phi}/\partial\boldsymbol{q})^T\boldsymbol{\varepsilon}\,, \tag{21}$$

where vector $\boldsymbol{\varepsilon} \in \mathcal{R}^m$ of the $\delta\boldsymbol{q}$ projection values onto the $(\partial\boldsymbol{\Phi}/\partial\boldsymbol{q})^T$ constrained subspace has to be determined. By substituting (21) into (19), we can write

$$\boldsymbol{\varepsilon} = -\left[(\partial\boldsymbol{\Phi}/\partial\boldsymbol{q})(\partial\boldsymbol{\Phi}/\partial\boldsymbol{q})^T\right]^{1}\boldsymbol{\Phi}(\hat{p})\,, \tag{22}$$

where $(\partial\boldsymbol{\Phi}/\partial\boldsymbol{q})(\partial\boldsymbol{\Phi}/\partial\boldsymbol{q})^T \in \mathcal{R}^{m\times m}$ is invertible under assumption that system constraints are independent and $(\partial\boldsymbol{\Phi}/\partial\boldsymbol{q})^T$ has full row rank.

By substituting (22) back to (21), we obtain final equation for determination of the needed correction values in the form [13]

$$\delta \boldsymbol{q} = -(\partial \boldsymbol{\Phi} / \partial \boldsymbol{q})^T \left[(\partial \boldsymbol{\Phi} / \partial \boldsymbol{q})(\partial \boldsymbol{\Phi} / \partial \boldsymbol{q})^T \right]^{-1} \boldsymbol{\Phi}(\hat{p}) = \begin{bmatrix} \delta \mathbf{x} \\ \delta \boldsymbol{\Psi} \end{bmatrix}. \tag{23}$$

Once we determine stabilization correction terms $\Delta\mathbf{x}$, $\Delta\boldsymbol{\Psi}$ from (23), a position correction for the linear displacement $\Delta\mathbf{x}$ has a final value for the step (variable $\mathbf{x}$ is also a 'global' coordinate of the manifold $\mathcal{S}$), while the final corrected value of the $\mathbf{R}_i \in SO(3)$ should be calculated on the basis of $\Delta\boldsymbol{\Psi}$ *via* relation

$$\mathbf{R}_i = \hat{\mathbf{R}}_i \exp(\Delta\tilde{\boldsymbol{\Theta}}_i) = \exp(\hat{\tilde{\boldsymbol{\Psi}}}_i + \Delta\tilde{\boldsymbol{\Psi}}_i). \tag{24}$$

It shold be emphasised that the proposed stabilization algorithm is straightforward and non-iterative, but it should be undertaken frequently during the integration process since the small correction values that can be substituted by the first variations are assumed.

3. NUMERICAL EXAMPLE

As an example of the case with large 3D rotations domain, a dual-spin satellite (also known as a gyrostat) is considered. The satellite is illustrated in Figure 1. It is composed of two rigid body connected by a revolute joint. There is usually large section, called the rotor (A) who contains satellite housekeeping equipment (solar arrays, main control computer and etc.) and platform (B) which usually contains the actual communications repeaters. The considered satellite is based on the model of the dual-spin satellite GOES-7. The mass and inertia tensor of the rotor are $300\,\mathrm{kg}$ and $\mathbf{J}_\mathrm{A} = diag(175,175,150)\,\mathrm{kg\,m^2}$. The platform's mass and inertia tensor yield $100\,\mathrm{kg}$ and $\mathbf{J}_\mathrm{B} = diag(43.75,43.75,50)\,\mathrm{kg\,m^2}$ respectively. The reference points and initial conditions are $\mathbf{X}_\mathrm{A} = [0 \quad 0 \quad 0]^T$, $\mathbf{X}_\mathrm{B} = [0 \quad 0 \quad 1.75]^T$, $\mathbf{R}_\mathrm{A} = \mathbf{I}$, $\mathbf{R}_\mathrm{B} = \mathbf{I}$, $\boldsymbol{\omega}_{\mathrm{A}0} = [0 \quad 0 \quad 1]^T\,\mathrm{rad/s}$ and $\boldsymbol{\omega}_{\mathrm{B}0} = [0 \quad 0 \quad 0.1]^T\,\mathrm{rad/s}$. Furhermore, a constant torque $\mathbf{M}_\mathrm{A}^a = [100 \quad 200 \quad 0]^T\,\mathrm{Nm}$ is applied to the rotor.

A matematical model for the dynamic simulation of satellite is shaped as a differential-algebraic system (DAE) of index 1. Translational and rotational parts of system dynamical equations are given in the standard form

$$m_\mathrm{A}\dot{\mathbf{v}}_\mathrm{A} - \mathbf{C}_\mathrm{A}^T\lambda^{if} = \mathbf{0}_{3\times1}, \tag{25}$$

$$m_\mathrm{B}\dot{\mathbf{v}}_\mathrm{B} - \mathbf{C}_\mathrm{B}^T\lambda^{if} = \mathbf{0}_{3\times1}, \tag{26}$$

$$\mathbf{J}_\mathrm{A}\dot{\boldsymbol{\omega}}_\mathrm{A} + \tilde{\boldsymbol{\omega}}_\mathrm{A}\mathbf{J}_\mathrm{A}\boldsymbol{\omega}_\mathrm{A} + \tilde{\mathbf{X}}_\mathrm{A}\mathbf{R}_\mathrm{A}^T\lambda^{if} = \mathbf{M}_\mathrm{A}^a, \tag{27}$$

$$\mathbf{J}_\mathrm{B}\dot{\boldsymbol{\omega}}_\mathrm{B} + \tilde{\boldsymbol{\omega}}_\mathrm{B}\mathbf{J}_\mathrm{B}\boldsymbol{\omega}_\mathrm{B} + \tilde{\mathbf{X}}_\mathrm{B}\mathbf{R}_\mathrm{B}^T\lambda^{if} = \mathbf{0}_{3\times1}, \tag{28}$$

where $\mathbf{x}_\mathrm{A}$ and $\mathbf{x}_\mathrm{B}$ are the rotor and platform mass center positions, $\boldsymbol{\omega}_\mathrm{A}$ and $\boldsymbol{\omega}_\mathrm{B}$ represents rotor and platform angular velocities, $m_\mathrm{A}, m_\mathrm{B}, \mathbf{J}_\mathrm{A}$ and $\mathbf{J}_\mathrm{B}$ are rotor and platform masses and tensors of inertia, λ^{if} stands for joint reaction force, $\mathbf{C}_\mathrm{A}$ and $\mathbf{C}_\mathrm{B}$ are rotor and platform constraint matrices, $\mathbf{X}_\mathrm{A}$ and $\mathbf{X}_\mathrm{B}$ are rotor and platform mass centers in the local coordinate system fixed to the bodys, and $\mathbf{R}_\mathrm{A}$ and $\mathbf{R}_\mathrm{B}$ are rotation matrices that relates body coordinate system to inertial coordinate system. Mechanical system constraints at position, velocity and acceleration level that represent joint between rotor and platform are given as

$$\mathbf{x}_\mathrm{A} + \mathbf{R}_\mathrm{A}\mathbf{X}_\mathrm{A} - \mathbf{x}_\mathrm{B} - \mathbf{R}_\mathrm{B}\mathbf{X}_\mathrm{B} = \mathbf{0}_{3\times1}, \tag{29}$$

$$\left[(\mathbf{R}_A\mathbf{E}_{A1})^T\mathbf{R}_B\mathbf{E}_{B3}\quad(\mathbf{R}_A\mathbf{E}_{A2})^T\mathbf{R}_B\mathbf{E}_{B3}\right]^T=\mathbf{0}_{2\times1}, \tag{30}$$

$$\begin{bmatrix}\mathbf{I}_3 & -\mathbf{R}_A\tilde{\mathbf{X}}_A & -\mathbf{I}_3 & \mathbf{R}_B\tilde{\mathbf{X}}_B\\ \mathbf{0}_{1\times3} & -(\mathbf{R}_B\mathbf{E}_{B3})^T\mathbf{R}_A\tilde{\mathbf{E}}_{A1} & \mathbf{0}_{1\times3} & -(\mathbf{R}_A\mathbf{E}_{A1})^T\mathbf{R}_B\tilde{\mathbf{E}}_{B3}\\ \mathbf{0}_{1\times3} & -(\mathbf{R}_B\mathbf{E}_{B3})^T\mathbf{R}_A\tilde{\mathbf{E}}_{A2} & \mathbf{0}_{1\times3} & -(\mathbf{R}_A\mathbf{E}_{A2})^T\mathbf{R}_B\tilde{\mathbf{E}}_{B3}\end{bmatrix}\begin{bmatrix}\mathbf{v}_A\\ \boldsymbol{\omega}_A\\ \mathbf{v}_B\\ \boldsymbol{\omega}_B\end{bmatrix}=\mathbf{0}_{5\times1}, \tag{31}$$

$$\begin{bmatrix}\mathbf{I}_3 & -\mathbf{R}_A\tilde{\mathbf{X}}_A & -\mathbf{I}_3 & \mathbf{R}_B\tilde{\mathbf{X}}_B\\ \mathbf{0}_{1\times3} & -(\mathbf{R}_B\mathbf{E}_{B3})^T\mathbf{R}_A\tilde{\mathbf{E}}_{A1} & \mathbf{0}_{1\times3} & -(\mathbf{R}_A\mathbf{E}_{A1})^T\mathbf{R}_B\tilde{\mathbf{E}}_{B3}\\ \mathbf{0}_{1\times3} & -(\mathbf{R}_B\mathbf{E}_{B3})^T\mathbf{R}_A\tilde{\mathbf{E}}_{A2} & \mathbf{0}_{1\times3} & -(\mathbf{R}_A\mathbf{E}_{A2})^T\mathbf{R}_B\tilde{\mathbf{E}}_{B3}\end{bmatrix}\begin{bmatrix}\dot{\mathbf{v}}_A\\ \dot{\boldsymbol{\omega}}_A\\ \dot{\mathbf{v}}_B\\ \dot{\boldsymbol{\omega}}_B\end{bmatrix}= \tag{32}$$

$$=\begin{bmatrix}-\mathbf{R}_A\tilde{\boldsymbol{\omega}}_A\tilde{\boldsymbol{\omega}}_A\mathbf{X}_A+\mathbf{R}_B\tilde{\boldsymbol{\omega}}_B\tilde{\boldsymbol{\omega}}_B\mathbf{X}_B\\ ((\mathbf{R}_B\tilde{\boldsymbol{\omega}}_B\mathbf{E}_{B3})^T\mathbf{R}_A\tilde{\mathbf{E}}_{A1}+(\mathbf{R}_B\mathbf{E}_{B3})^T\mathbf{R}_A\tilde{\boldsymbol{\omega}}_A\tilde{\mathbf{E}}_{A1})\boldsymbol{\omega}_A+((\mathbf{R}_A\tilde{\boldsymbol{\omega}}_A\mathbf{E}_{A1})^T\mathbf{R}_B\tilde{\mathbf{E}}_{B3}+(\mathbf{R}_A\mathbf{E}_{A1})^T\mathbf{R}_B\tilde{\boldsymbol{\omega}}_B\tilde{\mathbf{E}}_{B3})\boldsymbol{\omega}_B\\ ((\mathbf{R}_B\tilde{\boldsymbol{\omega}}_B\mathbf{E}_{B3})^T\mathbf{R}_A\tilde{\mathbf{E}}_{A2}+(\mathbf{R}_B\mathbf{E}_{B3})^T\mathbf{R}_A\tilde{\boldsymbol{\omega}}_A\tilde{\mathbf{E}}_{A2})\boldsymbol{\omega}_A+((\mathbf{R}_A\tilde{\boldsymbol{\omega}}_A\mathbf{E}_{A2})^T\mathbf{R}_B\tilde{\mathbf{E}}_{B3}+(\mathbf{R}_A\mathbf{E}_{A2})^T\mathbf{R}_B\tilde{\boldsymbol{\omega}}_B\tilde{\mathbf{E}}_{B3})\boldsymbol{\omega}_B\end{bmatrix}$$

where $\{\mathbf{E}_{A1},\mathbf{E}_{A2},\mathbf{E}_{A3}\}$ and $\{\mathbf{E}_{B1},\mathbf{E}_{B2},\mathbf{E}_{B3}\}$ are two triads of orthogonal unit vectors fixed to the rotor and platform. The system constraint matrix is shaped in the form

$$\mathbf{C}=\begin{bmatrix}\mathbf{I}_3 & -\mathbf{R}_A\tilde{\mathbf{X}}_A & -\mathbf{I}_3 & \mathbf{R}_B\tilde{\mathbf{X}}_B\\ \mathbf{0}_{1\times3} & -(\mathbf{R}_B\mathbf{E}_{B3})^T\mathbf{R}_A\tilde{\mathbf{E}}_{A1} & \mathbf{0}_{1\times3} & -(\mathbf{R}_A\mathbf{E}_{A1})^T\mathbf{R}_B\tilde{\mathbf{E}}_{B3}\\ \mathbf{0}_{1\times3} & -(\mathbf{R}_B\mathbf{E}_{B3})^T\mathbf{R}_A\tilde{\mathbf{E}}_{A2} & \mathbf{0}_{1\times3} & -(\mathbf{R}_A\mathbf{E}_{A2})^T\mathbf{R}_B\tilde{\mathbf{E}}_{B3}\end{bmatrix}, \tag{33}$$

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

$$\begin{bmatrix}m_A\mathbf{I}_3 & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_3 & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1}\\ \mathbf{0}_{3\times3} & \mathbf{J}_A & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \tilde{\mathbf{X}}_A\mathbf{R}_A^T & \tilde{\mathbf{E}}_{A1}\mathbf{R}_A^T\mathbf{R}_B\mathbf{E}_{B3} & \tilde{\mathbf{E}}_{A2}\mathbf{R}_A^T\mathbf{R}_B\mathbf{E}_{B3}\\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & m_B\mathbf{I}_3 & \mathbf{0}_{3\times3} & -\mathbf{I}_3 & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1}\\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{J}_B & -\tilde{\mathbf{X}}_B\mathbf{R}_B^T & \tilde{\mathbf{E}}_{B3}\mathbf{R}_B^T\mathbf{R}_A\mathbf{E}_{A1} & \tilde{\mathbf{E}}_{B3}\mathbf{R}_B^T\mathbf{R}_A\mathbf{E}_{A2}\\ \mathbf{I}_3 & -\mathbf{R}_A\tilde{\mathbf{X}}_A & -\mathbf{I}_3 & \mathbf{R}_B\tilde{\mathbf{X}}_B & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1}\\ \mathbf{0}_{1\times3} & -(\mathbf{R}_B\mathbf{E}_{B3})^T\mathbf{R}_A\tilde{\mathbf{E}}_{A1} & \mathbf{0}_{1\times3} & -(\mathbf{R}_A\mathbf{E}_{A1})^T\mathbf{R}_B\tilde{\mathbf{E}}_{B3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times1} & \mathbf{0}_{1\times1}\\ \mathbf{0}_{1\times3} & -(\mathbf{R}_B\mathbf{E}_{B3})^T\mathbf{R}_A\tilde{\mathbf{E}}_{A2} & \mathbf{0}_{1\times3} & -(\mathbf{R}_A\mathbf{E}_{A2})^T\mathbf{R}_B\tilde{\mathbf{E}}_{B3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times1} & \mathbf{0}_{1\times1}\end{bmatrix}\begin{bmatrix}\dot{\mathbf{v}}_A\\ \dot{\boldsymbol{\omega}}_A\\ \dot{\mathbf{v}}_B\\ \dot{\boldsymbol{\omega}}_B\\ \lambda^{if}\\ \lambda_1^{it}\\ \lambda_2^{it}\end{bmatrix}= \tag{34}$$

$$=\begin{bmatrix}\mathbf{0}_{3\times1}\\ -\tilde{\boldsymbol{\omega}}_A\mathbf{J}_A\boldsymbol{\omega}_A+\mathbf{M}_A^a\\ \mathbf{0}_{3\times1}\\ -\tilde{\boldsymbol{\omega}}_B\mathbf{J}_B\boldsymbol{\omega}_B\\ -\mathbf{R}_A\tilde{\boldsymbol{\omega}}_A\tilde{\boldsymbol{\omega}}_A\mathbf{X}_A+\mathbf{R}_B\tilde{\boldsymbol{\omega}}_B\tilde{\boldsymbol{\omega}}_B\mathbf{X}_B\\ ((\mathbf{R}_B\tilde{\boldsymbol{\omega}}_B\mathbf{E}_{B3})^T\mathbf{R}_A\tilde{\mathbf{E}}_{A1}+(\mathbf{R}_B\mathbf{E}_{B3})^T\mathbf{R}_A\tilde{\boldsymbol{\omega}}_A\tilde{\mathbf{E}}_{A1})\boldsymbol{\omega}_A+((\mathbf{R}_A\tilde{\boldsymbol{\omega}}_A\mathbf{E}_{A1})^T\mathbf{R}_B\tilde{\mathbf{E}}_{B3}+(\mathbf{R}_A\mathbf{E}_{A1})^T\mathbf{R}_B\tilde{\boldsymbol{\omega}}_B\tilde{\mathbf{E}}_{B3})\boldsymbol{\omega}_B\\ ((\mathbf{R}_B\tilde{\boldsymbol{\omega}}_B\mathbf{E}_{B3})^T\mathbf{R}_A\tilde{\mathbf{E}}_{A2}+(\mathbf{R}_B\mathbf{E}_{B3})^T\mathbf{R}_A\tilde{\boldsymbol{\omega}}_A\tilde{\mathbf{E}}_{A2})\boldsymbol{\omega}_A+((\mathbf{R}_A\tilde{\boldsymbol{\omega}}_A\mathbf{E}_{A2})^T\mathbf{R}_B\tilde{\mathbf{E}}_{B3}+(\mathbf{R}_A\mathbf{E}_{A2})^T\mathbf{R}_B\tilde{\boldsymbol{\omega}}_B\tilde{\mathbf{E}}_{B3})\boldsymbol{\omega}_B\end{bmatrix}$$

In equation (34), the multiplier λ^{if} is interpreted as joint reaction forces, whereas λ_1^{it} and λ_2^{it} are interpreted as joint reaction torques in the revolute joint along axis $\mathbf{E}_{A1}$ and $\mathbf{E}_{A2}$. The equation (34) is shaped as DAE of index 1 and integrated by the routine based on 4rd order explicit Munthe-Kaas algorithm. The results of numerical integration are given by the Figures 2 – 4.

Mathematical model is implemented in the MATLAB programming environment, while ADAMS package has been used only for post-processing based on the off-line calculation *via* described matematical model. The spatial trajectories of rotor and platform mass centres are given in Figure 2. The results shown in Figure 3 and 4 present components of the position of rotor and platform mass centres and angular velocitys.

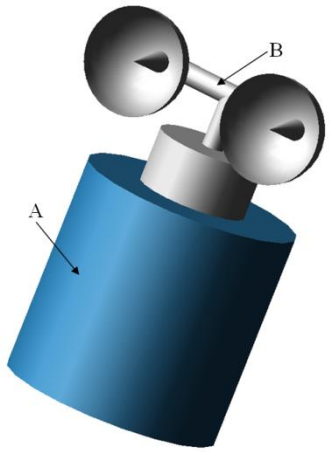

Figure 1. Sequence of the motion animation (post-processed via ADAMS).

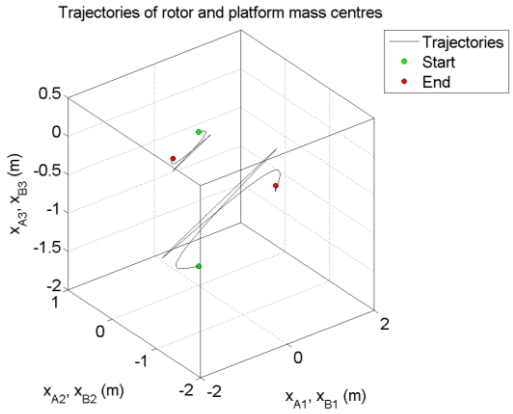

Figure 2. Spatial trajectories of rotor and platform mass centres.

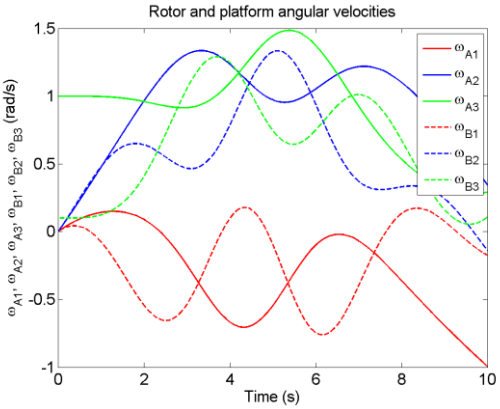

Figure 3. Rotor and platform angular velocities.

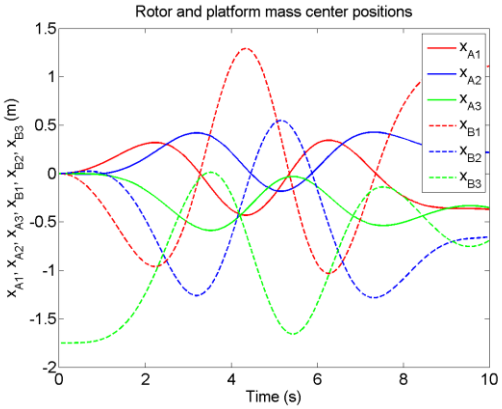

Figure 4. Rotor and platform mass center positions.

Proposed integration algorithm operates directly on the Lie-group of the system state space, avoiding kinematical singularities that are always present within the vector space formulations of the large 3D rotation domain, such as one presented here. By overviewing Figures 2–4 which are showing integral curves of the system position and angular velocity, it is visible that all obtained results are smooth functions without any discontinuities.

4. CONCLUSIONS

The Lie-group integration method for constrained multibody systems is proposed in the paper. The method operates on Lie-group of system configuration space that is modeled as 'state space formulation'. The system constraints are introduced in the mathematical model *via* DAE of index 1 formulation.

In order to stabilize constraint violation during integration procedure, two constraint stabilization algorithms are described: a constraint violation minimization by using constraint manifold projection methods based on solving nonlinear constrained least square problem (the algorithm is based on global

system coordinates and operates directly on the system state-space manifold) and projections along the constraint gradients by using (local) Cartesian vector rotation parameterization.
Since integration algorithm operates directly with angular velocities and rotational matrices, meaning that no local (generalized) coordinates are introduced, the method circumvent problems of kinematic singularities of rigid body three-parameters rotation basis, re-parameterization of system kinematics during integration as well as numerical non-efficiency of the kinematic differential equations. Within the presented example, method showed numerical robustness and it is easy-applicable on the general class of multibody systems.

ACKNOWLEDGEMENTS

The first and third author acknowledge the support of the Croatian Science Foundation under the contract of the project 'Geometric Numerical Integrators on Manifolds for Dynamic Analysis and Simulation of Structural Systems' that is conducted at Chair of Flight Vehicle Dynamics, Faculty of Mechanical Engineering and Naval Architecture, University of Zagreb.

REFERENCES

[1] Brüls, O. and Cardona, A.: On the Use of Lie Group Time Integrators in Multibody Dynamics. Journal of Computational and Nonlinear Dynamic, Vol. 5, 1–23, 2010.

[2] Cardona, A., Geradin, M.: A Beam Finite Element Non-Linear Theory with Finite Rotations. Int. Journal for Numerical Methods in Engineering, Vol. 26, 2403–2438, 1988.

[3] Celledoni, E., Owren, B.: Lie Group Methods for Rigid Body Dynamics and Time Integration on Manifolds. Comput. Methods Appl. Mech. Engrg., Vol. 192, 421–438, 2003.

[4] Hairer, E., Lubich, C. and Wanner, G.: Geometric Numerical Integration. Springer, 2006.

[5] Iserles, Munthe–Kaas, Norsett and Zanna: Lie-group methods. Acta Numerica, Vol. 9, 215–365, 2000.

[6] Makinen, J.: Critical study of Newmark-scheme on manifold of finite rotations. Computer Methods in Applied mechanics and Engineering, Vol. 191, 817–828, 2001.

[7] Marthinsen, A., Munthe–Kaas, H. and Owren, B.: Simulation of Ordinary Differential Equations on Manifolds: Some Numerical Experiments and Verifications. Modeling, Identification and Control, Vol. 18, 75–80, 1997.

[8] Müller, A.: Approximation of finite rigid body motions from velocity fields. Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik (ZAMM), Vol. 90, 514–521, 2010.

[9] Müller, A.: Group Theoretical Approaches to Vector Parameterization of Rotations. Journal of Geometry and Symmetry in Physics, Vol. 4, 1–30, 2005.

[10] Munthe-Kaas, H.: Lie-Butcher theory for Runge–Kutta methods. BIT 35, 572–587, 1995.

[11] Munthe-Kaas, H.: Runge–Kutta methods on Lie groups. BIT 38, 92–111, 1998.

[12] Terze, Z. and Naudet, J.: Geometric Properties of Projective Constraint Violation Stabilization Method for Generally Constrained Multibody Systems on Manifolds. Multibody System Dynamics, Vol. 20, 85–106, 2008.

[13] Yoon, S., Howe, R.M., Greenwood, D.T.: Geometric elimination of constraint violations in numerical simulation of Lagrangian equations. J. Mech. Des., Vol. 116, 1058–1064, 1994.