CHAINS OF RIGID BODIES AND THEIR NUMERICAL SIMULATION BY LOCAL FRAME METHODS

Nicolai Sætran
Nordre Krabbedalen 17
5178 Loddefjord, Norway

Antonella Zanna*
Matematisk Institutt
Universitetet i Bergen
5020 Bergen, Norway

Abstract. We consider the dynamics and numerical simulation of systems of linked rigid bodies (chains). We describe the system using the moving frame method approach of [18]. In this framework, the dynamics of the jth body is described in a frame relative to the (j−1)th one. Starting from the Lagrangian formulation of the system on SO(3)^N, the final dynamic formulation is obtained by variational calculus on Lie groups. The obtained system is solved by using unit quaternions to represent rotations and numerical methods preserving quadratic integrals.

1. Introduction. We consider the dynamics and numerical simulation of systems of linked rigid bodies (chains). We describe the chain system using the moving frame method (MFM) of [18] which is a Lagrangian variational approach that yields Newton-type equation. The approach is targeted to engineering students in order to give an elementary understanding of how to do constraint variations on Lie groups in order to build more complex dynamical systems using simple building blocks [10].

There are two main approaches to computational rigid body mechanics, a Lagrangian (variational) approach and a Newtonian approach. Most engineering applications use a Newtonian approach, formulating the problem as the Newton–Euler equations. In this approach, the main tool is Newton’s second law. Reaction forces have to be explicitly considered (which might be or might be not important, depending on the application), and the equations may result in a DAE systems, or ODE systems, provided that the reaction forces are embedded in the algorithm as intermediate variables [5]. If carefully implemented using a recursive algorithm, the resulting equations can be solved in O(N) operations, where N is the number of rigid bodies, see for instance [19, 5, 15, 4, 22]. This complexity was achieved without using the knowledge of Lie theory. Generally, the complexity of the algorithms depends strongly on the formulation and the choice of variables and coordinate systems to represent the system. For reasons that are discussed in great detail in the

2010 Mathematics Subject Classification. Primary: 70F10, 70E55, 65P99; Secondary: 65K99, 65L05.

Key words and phrases. N-body problems, chains of rigid bodies, variations on Lie groups, structure preservation.

* Corresponding author: Antonella Zanna.
relevant literature, most authors and practitioners in classical multibody dynam-ics follow the Newton–Euler approach and the recursive method, implemented in several research and industrial software packages.

Geometric Mechanics embraces instead the Lagrangian variational approach. In the Lagrangian approach, reaction forces are not needed because they are inherited by the formulation of the system. Holonomic constraints are dealt with either using Lagrange multipliers or using generalized coordinates, the minimal number of independent coordinates to describe the system.

Lie groups and Lie algebra methods have been used in the Lagrangian approach for several decades, see [16] for some history, and in the Newtonian approach to mechanics for at least three decades, see for instance the review article [17] and references therein.

The MFM is derived using the Lagrangian approach. It uses orthogonal matrices to describe rotations and skew-symmetric matrices to describe vector products of the type $\omega \times v$. The systems are constructed by adding a body at a time and the kinematic relations are obtained by describing each body orientation in the frame relative to the previous one. Conservative and non-conservative forces acting on the centers of mass are described in the inertial system, while torques are described in local body frames. The motion of each body is typically a rigid rotation or a translation.

The core of the MFM consists in relating the coordinates $X$ used to describe the system to the generalized coordinates $q$, which are the independent coordinates in which variations are done. This is accomplished through the introduction of the $B$ matrix, satisfying $\delta X = B\delta q$ and $\dot{X} = B\dot{q}$, and its derivative $\dot{B}$. Problems treated with this method so far consist of few bodies, the rotations being either planar or treated by successive single-angle rotations (Euler angles), see for instance [18, 6, 9]. The resulting expressions for $B$ and $\dot{B}$ are typically long expressions of sine and cosines and would typically be generated by a symbolic manipulator. However, beyond three or four joints, these expressions grow so large and unwieldy that even a powerful personal computer would have problems storing the symbolic expression for $B$ and $\dot{B}$ [20]. Another issue with the application of Euler angles is the presence of singularities in the equations of motion when studying rotations with three degrees of freedom (for instance the gimbal lock) and produce significant difficulties.

The goal of this paper is to formulate the MFM approach using the theory of Lie groups and Lie algebras\(^1\) to obtain a block matrix formulation for $B$ and $\dot{B}$ that: i) is amenable for spherical joints (and not only planar ones); and ii) is amenable for an arbitrary number $N$ of joints. Translations can also be considered in addition to rotations (prismatic joints) with some minor adjustments, as well as constraint rotations and tree-structures of rigid bodies, but, for sake of simplicity, in this paper we consider only chains of bodies with spherical joints.

Using computations on Lie groups and Lie algebras [12], the long expressions with sines and cosines are replaced by block matrices. Our preliminary experiments indicate that the cost of the approach is $O(N^2)$, which is common to many methods in mechanical engineering. An optimization of the algorithm has been outside the scope of the present paper. It can be interesting to investigate whether a recursive

\(^1\)Lie groups and Lie algebras are already employed in the formulation of the MFM, but only considering products of one-dimensional subgroups (one angle at a time).
approach that leads to $O(N)$ computations can be applied also to the proposed formulation.

The structure of the paper is as follows. In Section 2 we introduce the MFM of [18] and we show the variational derivation of the equation of motions in the general setting. We then describe the relation between the coordinates $X$ used to describe the systems and the choice of the generalized coordinates $q$ (rotation of frame $j$ relative to frame $j-1$). This relation is expressed as

$$\delta X = B\delta q, \quad \dot{X} = \dot{B}q,$$

for both the variations and the velocities. This relation is at the core of the MFM. In Section 3 we derive the systematic expressions for $B$ and $\dot{B}$ and formulate these as block matrices. We further introduce the quaternion notation for the rotations and derive the final form of the equations. Quaternions have also the advantage of removing the charts singularities due to the Euler angles. In Section 4 we show some numerical examples and applications to several type of problems and finally, in Section 5 we present some concluding remarks.

2. Background and notation. We consider a variational principle that allows for external and non conservative forces. Let $L(X, \dot{X})$ be the Lagrangian of the system, $L = K - U$, where $K$ is the kinetic energy and and $U$ is a potential, and let $W$ be the work done by external forces $F$ including non-conservative forces. Then the motion of the system from time $t_0$ to time $T$ is such that

$$\delta \int_{t_0}^{T} L(X, \dot{X}) dt = - \int_{t_0}^{T} \delta W dt$$

holds for infinitesimal variations consistent with the constraints \[7\].

2.1. The MFM: Deriving the equations of motion. In many engineering applications it is more convenient to consider conservative forces (like for instance gravity) as external forces and use only the kinetic energy $K$ as the Lagrangian. The Lagrange-d’Alembert principle $\delta \int_{t_0}^{T} L(X, \dot{X}) dt = - \int_{t_0}^{T} \delta W dt$ reduces to

$$\int_{t_0}^{T} \delta K dt = - \int_{t_0}^{T} \delta W dt, \quad K = \frac{1}{2} \dot{X}^{\top} M \dot{X},$$

where $\delta W$ is the virtual work of the forces in $F$.

The typical approach of the MFM is to use a set of coordinates

$$X = \begin{bmatrix} x \\ \theta \end{bmatrix}$$

where $x$ is the vector with the coordinates of the center of masses of the bodies in the inertial system, and $\theta$ represents one-dimensional rotation (the bodies can rotate only along a single given axis). As the bodies are rigid and linked to each other, these variables are not independent. Therefore, a central part to the derivation of the Euler-Lagrange equations in [18] relies on the fact that the velocities and virtual displacements are linearly related to the generalized velocity and generalized virtual displacement respectively. That is,

$$\delta X = B\delta q, \quad \dot{X} = \dot{B}q,$$
where now the \(q\) are independent variables (generalized coordinates) with no constraints. It is precisely the computation of the matrix \(B\) (and its derivative \(\dot{B}\)) that in the original approach consists in complicated expressions of sine and cosine functions and that here we shall construct in a more general manner.

As the variables are rotations, there are several ways to do variations. One way is to use constraints and Lagrange multipliers [7]. For Lie groups, which are usually parametrized using Lie-algebra coordinates as independent variables, the variations need to take into account the curvature of the underlying manifold, see for instance [14] for a thorough treatment of variations on manifolds.

For completeness, we summarize the derivation of the equations from [18], which uses the Lie-group formulation and the relations (1)-(2). The equations in [18] have been derived for one-parameter rotations, but it is straightforward to see the procedure extends in a similar manner to arbitrary rotations. We have

\[
\int_{t_0}^{T} \delta \dot{X}^\top M \dot{X} + \delta X^\top F dt = 0,
\]

(the mass-matrix \(M\) is symmetric). Since \(X\) contains both translational and rotational variables, we obtain

\[
\int_{t_0}^{T} (d/dt (\delta X) + D\delta X)^\top M \dot{X} + \delta X(t)^\top F dt = 0,
\]

where \(D\) is a block-diagonal matrix with zero blocks on the diagonal for the translational variables and skew-symmetric \(3 \times 3\) blocks for the angular variables. The matrix \(D\) corresponds to the \(\text{ad}\)-operator when doing variations on Lie groups, see [14]. Performing integration by parts and enforcing the boundary conditions, we obtain

\[
\int_{t_0}^{T} \delta X^\top \left( -\frac{d}{dt}(M \dot{X}) - DM \dot{X} + F \right) dt = 0.
\]

Next, using the linear relationships (1) and (2) between the \(\dot{X}\) and the generalized coordinates \(\dot{q}\),

\[
-\int_{t_0}^{T} \delta \dot{q}^\top B^\top \left( \frac{d}{dt} (MB\dot{q}) + DMB\dot{q} - F \right) dt = 0
\]

\[
-\int_{t_0}^{T} \delta \dot{q}^\top B^\top \left( MB\ddot{q} + (MB + DMB)\dot{q} - F \right) dt = 0.
\]

Imposing that the above holds for all possible variations \(\delta q\), we obtain the Euler-Lagrange equations:

\[
B^\top \left( MB\ddot{q} + (MB + DMB)\dot{q} \right) = B^\top F
\]

which we rewrite in the following form (Newton-type equations)

\[
M^* \ddot{q} + N^* \dot{q} = F^*,
\]

where

\[
\begin{align*}
M^* &= B^\top MB \\
N^* &= B^\top (MB + DMB) \\
F^* &= B^\top F
\end{align*}
\]

Note that \(B, \dot{B}\) and \(D\) depend themselves on the generalized coordinates, therefore \(M^*, N^*\) and \(F^*\) are time-dependent.
2.2. Coordinates and generalized coordinates. Following [18], the translational and angular velocity and virtual displacements are assembled in the vectors $\dot{X}, \delta X \in \mathbb{R}^{6N},$

\[
\dot{X} = \begin{bmatrix} v_1 \\ \vdots \\ v_N \\ \omega_1 \\ \vdots \\ \omega_N \end{bmatrix}, \quad \delta X = \begin{bmatrix} \delta x_1 \\ \vdots \\ \delta x_N \\ \eta_1 \\ \vdots \\ \eta_N \end{bmatrix}
\]

respectively. Similarly, the force vector is given as

\[
F = \begin{bmatrix} f_1 \\ \vdots \\ f_N \\ \tau_1 \\ \vdots \\ \tau_N \end{bmatrix},
\]

where $f_j$ forces apply to the center of mass the inertial frame (for instance gravity), while the $\tau_j$ are torques and are applied in the local frame.

The generalized coordinates

\[
q = \begin{bmatrix} q_1 \\ \vdots \\ q_j \\ \vdots \\ q_N \end{bmatrix},
\]

can include linear and rotational variables. Linear variables could be for instance the centers of mass of prismatic joints. The coordinates of body $j$ are resolved with respect to body $j-1$ (explaining the name moving frame method). This approach is also called mixed form representation in multibody system dynamics (see [17], pg. 56).

Similarly, the generalized velocity and virtual essential displacement of the generalized coordinates are assembled in a vector:

\[
\dot{q} = \begin{bmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{bmatrix}, \quad \delta q = \begin{bmatrix} \delta q_1 \\ \vdots \\ \delta q_n \end{bmatrix}
\]

where we abuse notation denoting everything as $\dot{q}$. It must be stressed for the translatative generalized coordinates, the $\dot{q}_j$ are the usual derivatives. In the case of rotational generalized coordinates (and in general Lie-group type variables), the $\dot{q}_j$ are trivialized derivative in the algebra (angular velocities), and not in the tangent space.

From this point, we assume that the generalized variables are $q_j \equiv R_{j,j-1}$, representing the relative rotation of body $j$ with respect to the frame of body $j-1$. The variable $q_j$ is naturally described by an element in the Lie group $\text{SO}(3)$ of
orthogonal matrices. We have

\[ \delta X = B \delta q \]
\[ \dot{X} = B \dot{q} \]

for a matrix \( B \) to be computed.

Having introduced the generalized coordinates, the variation of the velocities can be written in the form

\[ \delta \dot{X} = \frac{d}{dt} (\delta X) + D \delta X, \]

where

\[ D = \begin{bmatrix}
0 & & & \\
& \ddots & & \\
& & 0 & \hat{q}_1 \\
& & & \ddots \\
& & & & \hat{q}_N
\end{bmatrix} \]

(10)

the 0 denotes the \( 3 \times 3 \) zero matrix, and the hat-map represents the usual map that transforms a 3-dimensional vector to a \( 3 \times 3 \) skew-symmetric matrix describing the exterior vector product [12, 14]. Note that we are abusing notation, as the \( \dot{q}_j \) will be angular velocities.

With the above definition of the \( X \), the kinetic energy of the system is easily expressed by the quadratic form:

\[ K(t) = \frac{1}{2} \dot{X}^\top M \dot{X}, \]

(11)

where \( M \) is the symmetric positive definite block-matrix

\[ M = \begin{bmatrix}
m_1 I & & \\
& \ddots & \\
& & m_N I \\
& J_1 & \\
& & \ddots \\
& & & J_N
\end{bmatrix}, \]

(12)

where \( m_j \) is the mass of body \( j \), \( I \) is the \( 3 \times 3 \) identity and the \( J_j \)s are the moment of inertia tensors relative to the fixed body frame.

The virtual work is given by

\[ \delta W(t) = \delta X(t)^\top F. \]

(13)

In the MFM setting, the choice of generalized coordinates is \( q_j \equiv R_{j,j-1} \), the rotation of frame \( j \) relative to frame \( j - 1 \). These are parametrized with the help of the \( \omega_{j,j-1} \in \mathbb{R}^3 \), so that

\[ \dot{q}_j \Rightarrow \omega_{j,j-1} \]
\[ \delta q_j \Rightarrow \eta_{j,j-1} \]
(abusing notation). The system evolves on SO(3)^N, N copies of the group of rotation matrices. Euclidean transformations are used to keep track of the center of mass of the bodies for the application of the external forces.

3. The B and \( \dot{B} \) matrices.

3.1. Computing \( B \). Let \( x_j \) be the position of the \( j \)th center of mass at time \( t \) in the inertial system. We denote by \( s_{j,1} \) the vector determining center of mass \( j \) (\( cm_j \)) from the end of body \( j - 1 \) and \( s_{j,2} \) the vector from \( cm_j \) to body \( j + 1 \), see Fig. 1. Both \( s_{j,1} \) and \( s_{j,2} \) are in local coordinates and are fixed in the \( j \)th body system. For the positions of the centers of mass in the inertial system, we have

\[
\begin{align*}
x_1 &= R_1 s_{1,1} \\
x_j &= x_{j-1} + R_{j-1} s_{j-1,2} + R_j s_{j,1}, \quad j = 2, \ldots, N,
\end{align*}
\]

where \( R_j \) is the overall rotation of body \( j \) with respect to the inertial system. Thus,

\[
\begin{align*}
R_1 &= R_{1,0} \\
R_j &= R_{1,0} R_{2,1} \cdots R_{j,j-1} = R_{j,0},
\end{align*}
\]

where \( R_{j,j-1} \) is the relative rotation of frame \( j \) with respect to frame \( j - 1 \).

Furthermore, we denote by

\[
R_{k,j} = R_{j+1,j} R_{j+2,j+1} \cdots R_{k,k-1} = R_j^T R_k, \quad R_{k,k} = I,
\]

the product of the relative rotation matrices from body \( j \) to body \( k \geq j \), that can be interpreted as the rotation of body \( k \) relative to the frame fixed with body \( j \).

Locally, we have \( \dot{R}_{j,j-1} = \hat{\omega}_{j,j-1} \hat{R}_{j,j-1} \), therefore, for the global rotation matrices, we have

\[
\dot{R}_j = R_j \left( \sum_{i=1}^{j} \text{Ad}_{R_{j,i}} \hat{\omega}_{i,i-1} \right),
\]

where

\[
\text{Ad}_{R_{j,i}}(\xi) = R_{j,i} \xi R_{j,i}^{-1}
\]
is a linear operator in $\xi$, see [12]. Identifying the right hand sides above with the right hand sides of $\dot{R}_j = R_j \hat{\omega}_j$, we see that

$$\hat{\omega}_j = \sum_{i=1}^{j} \text{Ad}_{R_{j,i}} \hat{\omega}_{i,i-1}. $$

Since $\text{Ad}_Q(\cdot)$ is linear in its argument and moreover

$$\text{Ad}_Q \hat{u} = \hat{Q}u$$

for any $3 \times 3$ orthogonal matrix $Q$ [12], the above relation becomes

$$\omega_j = \sum_{i=1}^{j} R^T_{j,i} \omega_{i,i-1}.$$

Setting up together, we can write

$$\omega = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \vdots \\ \omega_N \end{bmatrix} = \begin{bmatrix} I & R^T_2,1 \\ R^T_2,1 & I & \ddots \\ & \ddots & \ddots \\ & & R^T_N,1 & \cdots & R^T_{N,N-1} & I \end{bmatrix} \begin{bmatrix} \omega_{1,0} \\ \omega_{2,1} \\ \omega_{3,2} \\ \vdots \\ \omega_{n,n-1} \end{bmatrix} = LR\hat{q},$$

(15)

where $L_R$ is the lower triangular matrix relating the angular velocities in the inertial frame to those in the local frames. Introducing the block matrices

$$R = \text{diag}(R_1, R_2, \ldots, R_N),$$

and

$$\hat{\Omega} = \text{diag}(\hat{\omega}_1, \ldots, \hat{\omega}_N) = \text{diag}(\omega)$$

(17)

(where the latter hat-map is intended blockwise), we have

$$\hat{R} = R\hat{\Omega}, \quad \Omega = L_R\hat{q}.$$ (18)

To relate the velocities $v_j$ to the $q_j \equiv \omega_{j,j-1}$, we observe that

$$\dot{x}_j = \dot{x}_{j-1} + \dot{R}_{j-1}s_{j-1,2} + \dot{R}_j s_{j,1}.$$ (19)

In particular, for $x_1$, we have

$$\dot{x}_1 = \dot{R}_1 s_{1,1} = R_1 \hat{\omega}_{1,0} s_{1,1} = -R_1 \hat{s}_{1,1} \omega_{1,0}.$$ (19)

For the general term, we have already computed the $\dot{R}_j$. Furthermore, to make the above relation linear in the $\omega_j$s, we use (14) together with $\hat{u}v = -\hat{v}u$, to obtain

$$\dot{x}_j = \dot{x}_{j-1} + R_{j-1} \left( \sum_{i=1}^{j-1} \text{Ad}_{R_{j,i}} \hat{\omega}_{i,i-1} s_{j-1,2} \right) + R_j \left( \sum_{i=1}^{j-1} \text{Ad}_{R_{j,i}} \hat{\omega}_{i,i-1} s_{j,1} \right)$$

$$= \dot{x}_{j-1} + R_{j-1} \left( \sum_{i=1}^{j-1} R_{j,i} \hat{\omega}_{i,i-1} \right) s_{j-1,2} + R_j \left( \sum_{i=1}^{j-1} R_{j,i} \hat{\omega}_{i,i-1} \right) s_{j,1}$$

$$= \dot{x}_{j-1} - R_{j-1} \hat{s}_{j-1,2} \left( \sum_{i=1}^{j-1} R_{j,i} \hat{\omega}_{i,i-1} \right) - R_j \hat{s}_{j,1} \left( \sum_{i=1}^{j-1} R_{j,i} \hat{\omega}_{i,i-1} \right)$$

We introduce the following block diagonal matrices

$$\hat{S}_1 = \text{diag}(\hat{s}_{1,1}, \hat{s}_{2,1}, \ldots, \hat{s}_{N,1}), \quad \hat{S}_2 = \text{diag}(\hat{s}_{1,2}, \hat{s}_{2,2}, \ldots, \hat{s}_{N,2})$$

(19)
(see Fig. 1), and the block shift matrix
\[
\Delta_\omega = \begin{bmatrix}
0 & \cdots & 0 \\
I & 0 & \\
0 & \ddots & \ddots \\
\vdots & \ddots & \ddots \\
0 & \cdots & 0 & I \\
\end{bmatrix}
\]
(20)

We have the relation
\[
\dot{x} - \Delta_\omega \dot{x} = -(\Delta_\omega R\hat{S}_2 + R\hat{S}_1)L_R\dot{q},
\]
(21)
The matrix \(I_{3N} - \Delta_\omega\) is invertible,
\[
E_0 = (I_{3N} - \Delta_\omega)^{-1} = \begin{bmatrix}
I & 0 & \cdots & 0 \\
I & I & \\
\vdots & \ddots & \ddots \\
I & \cdots & I & I \\
\end{bmatrix} = I_{3N} + E_{-1}
\]
(22)
moreover, we observe that
\[
E_0 \Delta_\omega = E_{-1} = \begin{bmatrix}
0 & 0 & \cdots & 0 \\
I & 0 & \\
I & \ddots & \ddots \\
\vdots & \ddots & \ddots & 0 \\
I & \cdots & I & I \\
\end{bmatrix},
\]
thus, we have
\[
v = \dot{x} = -(E_{-1}R\hat{S}_2 + E_0R\hat{S}_1)L_R\dot{q}
\]
(23)
In compact form,
\[
\begin{bmatrix}
v \\
\omega
\end{bmatrix} \begin{bmatrix}
-E_{-1}R\hat{S}_2 - E_0R\hat{S}_1 \\
I_{3N}
\end{bmatrix} L_R\dot{q} = \begin{bmatrix}
B^{\text{transl}} \\
B^{\text{rot}}
\end{bmatrix} \dot{q} = B\dot{q}
\]
(24)
where
\[
B^{\text{transl}} = -(E_{-1}R\hat{S}_2 + E_0R\hat{S}_1)L_R, \quad B^{\text{rot}} = L_R.
\]

3.2. The \(\dot{\Omega}\) matrix. By direct differentiation, we have
\[
\dot{B}^{\text{transl}} = -(E_{-1}\dot{R}\hat{S}_2 + E_0\dot{R}\hat{S}_1)L_R - (E_{-1}R\hat{S}_2 + E_0R\hat{S}_1)\dot{L}_R \quad \dot{B}^{\text{rot}} = \dot{L}_R.
\]
From above, we have \(\dot{R} = R\hat{\Omega}\). Then, the most difficult part is to find \(\dot{L}_R\). Recall that the matrix \(L_R\) is made of blocks of the form \(R_{j,i}^T = (R_{i+1,i} \cdots R_{j,j-1})^T, j > i\).

We observe that
\[
R_{j,i} = R_{i+1,i} \cdots R_{j,j-1} = R_j^T R_i, \quad R_{j,i}^T = R_j^T R_i,
\]
therefore
\[
R_{j,i}^T R_i + R_j^T R_i = (R_j \hat{\omega}_j)^T R_i + R_j^T R_i \hat{\omega}_i = -\hat{\omega}_j R_j^T R_i + R_j^T R_i \hat{\omega}_i = R_{j,i}^T \hat{\omega}_i - \hat{\omega}_j R_{j,i}^T.
\]
Recalling the definition of \(\hat{\Omega}\), we recognize that the above equations can be written in matrix form as
\[
\dot{L}_R = L_R\hat{\Omega} - \hat{\Omega}L_R,
\]
(25)
a strictly lower triangular block matrix. Alternatively, one can notice that
\[ \dot{R}_{j,i} = R_{j,i} \hat{\omega}_j - \omega_i R_{j,i} \]
\[ = R_{j,i} (\hat{\omega}_j - R^T_{j,i} \hat{\omega}_i R_{j,i}) \]
\[ = R_{j,i} \hat{\omega}_{j,i} \]
where \( \hat{\omega}_{j,i} \) can be intended as the angular velocity for body \( j \) relative to the body frame \( i \), as \( R_{j,i} \) is the rotation matrix of body \( j \) relative to the body frame \( i \), so that
\[ \dot{B}_{\text{rot}}^{\text{rot}} = -\hat{\omega}_{j,i} R^T_{j,i}. \]

In summary, we have
\[ \dot{B} = \left[ \begin{array}{c} \dot{B}_{\text{trans}} \end{array} \right] = \begin{bmatrix} -((E_{-1} R \hat{S} \hat{S}_2 + E_0 R \hat{\Omega} \hat{S}_1) L_R - (E_{-1} R \hat{S} \hat{S}_2 + E_0 R \hat{S}_1) (L_R \hat{\Omega} - \hat{\Omega} L_R) \\ L_R \hat{\Omega} - \hat{\Omega} L_R \end{bmatrix} \]
\[ = (28) \]

3.3. Setting up the equations using the quaternion formalism. As mentioned before, the generalized coordinates \( q_j \equiv R_{j,j-1} \) are the relative rotations of body \( j \) relative to the body frame \( j - 1 \). To represent the rotations, we use unit quaternions. Letting \( \mathbb{H} \) being the set of quaternions, we abuse notation and identify
\[ R_{j,j-1} = q_j \in \mathbb{H}, \quad q_j = \begin{bmatrix} q_{j,0} \\ q_{j,v} \end{bmatrix}, \quad |q_{j,0}|^2 + ||q_{j,v}||^2 = 1 \]
where \( q_{j,0} \in \mathbb{R} \) and \( q_{j,v} \in \mathbb{R}^3 \) are the scalar and the vector components of the unit quaternion, respectively. Unit quaternions obey the differential equation
\[ \dot{q}_j = \frac{1}{2} q_j * \xi_j, \quad \xi_j \in \mathbb{H}, \quad \xi_{j,0} = 0, \quad \xi_{j,v} = \omega_{j,j-1} \equiv \dot{q}_j, \]
where \( * \) represents the quaternion product and \( \xi_{j,v} = \omega_{j,j-1} \equiv \dot{q}_j \in \mathbb{R}^3 \) is the angular velocity vector. The relative rotations \( q_j \equiv R_{j,j-1} \) can be reconstructed using the formulas
\[ R_{j,j-1} = I + 2q_{j,0} \hat{q}_{j,v} + 2q_{j,v}^2. \]
Writing
\[ q_j = \begin{bmatrix} q_{j,0} \\ q_{j,v} \end{bmatrix} = \begin{bmatrix} \cos(\frac{\theta_j}{2}) \\ \sin(\frac{\theta_j}{2}) n \end{bmatrix} \]
(27)
we see that the quaternion \( q_j \) can be also interpreted as a rotation by an angle \( \theta_j \) around the axis determined by the direction \( n \in \mathbb{R}^3 \).

Recall that the quaternion product can be expressed in terms of matrix-vector products, in particular, if \( r = [r_0, r_v]^T \in \mathbb{H}, \quad p \in \mathbb{H}, \) one can write
\[ r * p = Q(r)p, \quad Q(r) = r_0 I_4 + \begin{bmatrix} 0 & -r_v^T \\ r_v & \hat{r}_v \end{bmatrix}. \]
(28)
We denote
\[ q = \begin{bmatrix} q_1 \\ \vdots \\ q_N \end{bmatrix} \in \mathbb{H}^{4N}, \quad \xi_v = \begin{bmatrix} \xi_{1,v} \\ \vdots \\ \xi_{N,v} \end{bmatrix} = \begin{bmatrix} \omega_{1,0} \\ \vdots \\ \omega_{N,N-1} \end{bmatrix} = \dot{q} \in \mathbb{R}^{3N} \]
and introduce the block-diagonal matrix
\[ Q = \text{diag}(Q(q_1), \ldots, Q(q_N)) \]
where $Q(q_j)$ is as in (28). In the quaternion variables, the system (5) reads

$$
\dot{q} = \frac{1}{2} Q \xi \\
M^* \dot{\xi} + N^* \xi = F^*
$$

(29)

with $M^*, N^*, F^*$ as in (6).

We observe that the matrix $M^*$ is symmetric positive, since $M^* = B^T MB$, where the lower block of $B$ is $L_R$ which has identities blocks on the diagonal and is therefore full rank. This implies that $B$ is full rank, therefore the map $y \mapsto By$ is an injective map, mapping nonzero vectors to nonzero vectors. This implies that

$$
y^T M^* y = y^T B^T MB y = (By)^T M (By) > 0 \quad \forall y \neq 0,
$$

since $M$ is positive definite.

It follows that $M^*$ is invertible, and introducing the variable

$$
y = \begin{bmatrix} q \\ \xi v \end{bmatrix}
$$

the system can be written as a first order system

$$
\dot{y} = f(y) = \begin{bmatrix} \frac{1}{2} Q \xi \\ (M^*)^{-1} (F^* - N^* \xi) \end{bmatrix}.
$$

(30)

4. Application to the numerical integration of chains of rigid bodies.

The advantages of the quaternion formulation is that: 1) it reduces the storage compared to using rotation matrices directly, 2) the unitarity of the quaternion is a quadratic conservation law, therefore it can be preserved by symplectic methods [21, 8] (although this holds also for rotation matrices). Methods that do not preserve quadratic conservation laws would require a projection step or a very small stepsize to reduce the local error hence the unitarity error. Alternatively, one could use methods for orthogonal flows [11, 2, 13] or with Lie-group methods [3, 12, 1] but this is beside the scope of the present paper.

As a proof of concept, we will use Gauss–Legendre Runge–Kutta methods, which are implicit. To justify the choice, Table 1 shows a comparison between the 2-stages GLRK (order 4) and RK45 for a four-body pendulum in §4.3 only under the effect of gravity (no springs/dampers). The RK45 method is implemented with relative error tolerance $\text{rtol} = 1e-13$ and absolute tolerance $\text{atol} = 1e-13$. The GLRK2 is implemented with a stepsize $h = 0.01$ and a tolerance $\text{rtol} = 1e-9$ on the solution of the implicit iterations (by fixed point). The solutions of the two methods are compared at time $T = 10$.

The cost of the algorithm will depend on the number $N$ of links. Table 2 gives the timings of a $N$-body pendulum displaced by $\theta_0 = 0.1$ about the $x$-axis in the first body. The system is integrated up to $T = 10$ with a 3-stages GLRK (order
6) with stepsize $h = 0.01$. The computational cost grows quadratically with the number of links $N$. This is reasonable, as the matrix $L_R$ increases by a block-row at each time. Although we only add a rotation, we need also the relative rotations with respect to each of the previous $N - 1$ bodies.

In what follows, the numerical simulations are performed both in Matlab and Python. In Matlab, the reference solutions are computed using `ode45`, setting the relative tolerance to $\text{RelTol} = 2.22045\times10^{-14}$ and absolute tolerance to $\text{AbsTol} = 1\times10^{-14}$. In Python, the reference solutions are computed using the built-in package `scipy.integrate` and the method `solve_ivp`, using an explicit RK45 method with relative error tolerance $\text{rtol} = 1\times10^{-13}$ and absolute tolerance $\text{atol} = 1\times10^{-13}$.

### 4.1. Solid pendulum.

A pendulum shaped as a long narrow rod is fixed at point $O$ (origin of the inertial system) and free to rotate. At rest position, the center of mass is located at $l/2e_3$ in the inertial frame ($l_{cm} = l/2$). If the pendulum is displaced from the rest position by an angle $\theta_0$ and released (with zero initial velocity), the motion is planar and consists of rotation by an angle $\theta$ in the $yz$-plane (vertical plane, rotation about the $x$ axis). Using a classical parameterization by an angle $\theta$ from the vertical position, the equations of the motion of the pendulum are given as

$$J_O \ddot{\theta} = -mg \frac{l}{2} \sin \theta$$

where $m$ is the mass of the pendulum, $g$ the gravity constant and $J_O$ is the moment of inertia about the origin, $J_O = \frac{1}{3}ml^2$.

| $N$  | Time   |
|------|--------|
| 4    | 719ms±4.31ms |
| 8    | 2.27s±15.8ms  |
| 16   | 10.7s±122ms   |
| 32   | 45.6s±1.55s   |
| 48   | 1m43s±1.94s   |
| 64   | 3m24s±5.52s   |

**Table 2.** Computational times for systems with $N$ links. The computational cost grows quadratically with the number of links. See text for more details.

![Figure 2](image-url)

**Figure 2.** Single solid pendulum, consisting of a rectangular prism. Left: classical planar motion, described by the angle $\theta$. Right: planar motion, described in 3D by a quaternion and the MFM. At rest (vertical position), the inertial and frame axes are aligned.
Figure 3. Numerical solution for the angle $\theta$ for the solid pendulum. The MFM is solved with IMR and stepsize $h = 0.01$ and the exact solution is computed with RK45 imposing machine precision on the tolerances.

In the MFM, the $B$, $\dot{B}$, $D$ and $M$ matrices are trivial to formulate for this example,

$$ B = \begin{bmatrix} -R_1\hat{s}_{1,1} & I \end{bmatrix}, \quad \dot{B} = \begin{bmatrix} -R_1\hat{\omega}_1\hat{s}_{1,1} & 0 \end{bmatrix}, \quad M = \begin{bmatrix} mI & 0 \\ 0 & J \end{bmatrix}, \quad D = \begin{bmatrix} 0 & 0 \\ 0 & \hat{\omega}_1 \end{bmatrix} $$

where now $J$ is the matrix of the principal moments of inertia in the body frame,

$$ J = \begin{bmatrix} \frac{1}{2}m l^2 & 0 & 0 \\ 0 & \frac{1}{2}m l^2 & 0 \\ 0 & 0 & \varepsilon \end{bmatrix} $$

where $\varepsilon$ is a small number (we assume that the width of the prism is very small compared to its length $l$). In this set up, the force is

$$ F = \begin{bmatrix} f_1 \\ \tau_1 \end{bmatrix}, \quad f_1 = -m g e_3, \quad \tau_1 = 0, $$

as there is no external torque. The vector $y$ in (30) is simply

$$ y = \begin{bmatrix} q_1 \\ \xi_{1,v} \end{bmatrix} = \begin{bmatrix} q \\ \xi_v \end{bmatrix}, \quad \xi_v = \omega_{1,0} = \omega_1, $$

where $q$ is the rotation quaternion and $\omega_1 = \omega_{1,0}$ is the angular velocity of the body relative to the inertial system. Writing down explicitly the differential equation for $\omega_1$ using (30) we see that these are equivalent to

$$ (-m \hat{s}_{1,1}^2 + J)\dot{\omega}_1 = -(R_1^T f_1) \times s_{1,1} - J\hat{\omega}_1 \omega_1 = -(R_1^T f_1) \times s_{1,1} $$

$$(\hat{\omega}_1 \omega_1 = \omega_1 \times \omega_1 = 0).$$

We recognize the parallel axis theorem on the left hand side and the torque exerted by gravity on the center of mass on the right hand side, thus obtaining a vector version of (31).

In Figure 3 we compare the solution of the MFM with the exact solution $\theta$ from (31). For the MFM we use the Implicit Midpoint Rule (IMR, Gauss-Legendre method of order 2) implemented with stepsize $h = 0.01$. The angle $\theta$ is reconstructed as $\theta = 2 \arccos(q_{01})$, see (27). The numerical error is as expected from theory. The numerical solution recovers well the planar motion although the simulation is in full 3D. Clearly, the method is an overkill for this simple planar example, however, it shows how straightforward it is to set up full 3D computations with much more complicated forces and torques by merely changing $f_1$ and $\tau_1$ in (33).

For the simulations, we have used $l = 2$ m, and $m = 50$ kg and initial condition $\theta_0 = 0.1$. 
4.2. Spinning top. To illustrate the ease of the implementation of the method, we consider the case of a symmetric heavy top. We consider a top idealized as a solid rod (radius \( r_{\text{rod}} = 0.05 \text{m}, \ h_{\text{rod}} = 0.4 \text{m} \)) with a solid disk on top (radius \( r_{\text{disk}} = 0.5 \text{m}, \ h_{\text{disk}} = 0.05 \text{m} \)) both of constant density \( \rho = 1000 \text{kg/m}^3 \). The total mass of the body is \( m \approx 28.274 \text{kg} \), the center of mass is situated at 0.4m along the main symmetry axis of the body and the inertia matrix in body coordinates centered at the center of mass is

\[
J = \begin{bmatrix}
1.195768703772615 & 0 & 0 \\
0 & 1.195768703772615 & 0 \\
0 & 0 & 2.01456289114456
\end{bmatrix}.
\]

The inertia matrix is obtained by considering the individual inertia matrices of the rod and the disk, which are then translated to the center of mass using the parallel axis theorem. The form of the matrices \( B, \dot{B}, M, D \) for the system are precisely those of the pendulum (32), and so are the forces (33). The only difference is in the initial conditions and the parameter values. We use \( q_0 = [\cos(\theta_0/2), \sin(\theta_0/2)n] \), \( \theta_0 = \pi/6 \), \( n = [1, 0, 0]^\top \) and \( \xi_{r,0} = [0, 0, 25]^\top \) (corresponding to \( \dot{\theta}_0 = 25 \)). A simulation with these parameter \( (T = 30, h = 0.01) \) can be found in Figure 4, an animation, with slightly different parameters, can be found here: [https://youtu.be/cOL68IbniPE](https://youtu.be/cOL68IbniPE).

![Figure 4. A simulation of the spinning top, idealized as a rod with a disk (left). Right: position of the center of mass in [0, 30] integrated with step size h = 0.01. See text for the values of the remaining parameters. The formulation of the equations is identical to that of the single pendulum.](image)

4.3. 4-body pendulum. We consider a 4-body chain with torsional springs, located at the joints, which resist bending. The neutral position of the system (uncompressed springs) is the vertical upright orientation for all the links.

Let \( u = [\theta_j, \phi_j, \psi_j]^\top \) be the rotational displacement away from the neutral position of body \( j \) (in body coordinates). These displacements are computed by converting the rotation matrix \( R_{j-1,j} \) into angle displacements using the relation \( R = \exp(\hat{u}) \). Then \( \hat{u} = \log R = \frac{\sin^{-1}(\|y\|)}{\|y\|} \hat{y} \), where \( \hat{y} = \frac{1}{2}(R - R^\top) \), see [12]. Each displacement will result in a torsional force applied to body \( j \) and a corresponding
Figure 5. 4-body pendulum with torsional springs

\[ \tau_j = -k_j u = -k_j \begin{bmatrix} \theta_j \\ \phi_j \\ \psi_j \end{bmatrix}, \quad \tau_{j+1} = -R_{j+1,j}^T \tau_j \]

where \( k_j \) is the torsional spring coefficient. The spring coefficients are \( k_1 = 4000 \frac{kg.m}{s^2} \), \( k_2 = 3000 \frac{kg.m}{s^2} \), \( k_3 = 2000 \frac{kg.m}{s^2} \) and \( k_4 = 1000 \frac{kg.m}{s^2} \). Initially, the bodies are rotated first about the first axis and then the second axis by the same angle \( \theta_0 \),

\[
q_1(t_0) = q_2(t_0) = q_3(t_0) = q_4(t_0) = \begin{bmatrix} \cos(\theta_0) \\ \sin(\theta_0) \end{bmatrix}^T
\]

The initial angular velocities are zero,

\[ \xi_j(t_0) = 0, \quad j = 1, \ldots, 4. \]

The system is integrated with a 3-stages GLRK for \( t_0 = 0 < t_1 < \cdots < t_N = T \) with \([t_0, T] = [0, 15]\) and time step \( h = 0.01 \). An animation of the system solution is available at https://youtu.be/EA6zu04TB7I

4.4. A 16-body pendulum. This system consists of 16 linked bodies where the first link is hinged to the origin \( O \) of the inertial system. The hinges are all spherical, allowing the links to rotate in any direction.

The dynamic system results in a a 122-term vector ODE system evolving on \( SO(3)^{16} \).

Figure 6. 16-body
The initial condition for the first body is \( q_1(t_0) \) expressed by (27) with \( \theta_0 = \frac{\pi}{8} \) and the remaining coordinates without any displacement. The results from integration with the 2-stages GLRK method are displayed in Figure 7. As we expected, the quaternion unit length is preserved to the level of machine accuracy (GLRK method preserve quadratic integrals). Similarly also the relative error of the system is very well preserved in the numerical simulation.

Several types of external forces may be applied to the model. Let us consider a generic number of links \( N \). We consider a spring-damper with neutral length \( l \). The spring is attached at a point \( P \) situated at the end of body \( N \), with coordinates \( x_p = x_N + R_N s_{N,2} \) in the inertial frame, \( x_N \) being the coordinates of the \( N \)th center of mass in the inertial frame. Furthermore, let \( x_{cp} \) be the coordinates of the attachment of the other end of the spring to the inertial system. Let \( s = x_{cp} - x_p \), so that \( \sigma = ||s|| - l \) is the stretch of the spring and \( n = s/||s|| \) is the unit vector in the direction of the displacement. The spring force is then \( f_s = -k_s \sigma n \). Similarly, the damping is given as \( f_d = -k_d (v_p n) n \). The constants \( k_s \) and \( k_d \) are the spring and the damper coefficients, respectively.

This force \( f_s \) and the damping \( f_d \) generate a moment (torque) on the \( N \)th body. In local coordinates, this torque is given as \( \tau_N = s_{N,2} \times R_N^T f \), \( f = f_s + f_d \). The
vector $F$ looks like

$$F = \begin{bmatrix} -m_1 g e_3 \\ \vdots \\ -m_N g e_3 + f \\ 0 \\ \vdots \\ s_{N,2} \times R_N^T f \end{bmatrix}$$

An animation of this system can be found at \url{https://youtu.be/a_ibo_un0og}.

4.5. **A 64-body pendulum.** As an example of a significantly longer system is a 64-body system. The first body is hinged to the origin of the inertial system, and every subsequent body is hinged at the end of the previous body and free to rotate in any direction. The dynamic system is a 672-term vector ODE system generating a solution on $\text{SO}(3)^{16}$. The pendulum is coiled in the $xy$-plane and released under the effect of gravity. Torsional dampers are attached at the joints to reduce oscillations due to the stiffness of the system. We use the same body parameters and inertia moments as for the previous experiment, the joint-damper coefficients are $70\text{kg}$-$\text{s}$. An animation of the 64-body pendulum falling from rest is available at \url{https://youtu.be/efhnb3jQkas}.

![Figure 9. 64-body pendulum](image)

5. **Conclusions.** In this paper we have generalized the MFM of [18] to arbitrary systems of linked rigid bodies, allowing the bodies to rotate in all three dimensions. Further, we have systematized the way to construct the $B$ and the $\dot{B}$ matrices as block matrices for numerical simulations. The formulation does not require any form of symbolic manipulation as the matrices $B$ and $\dot{B}$ can be computed on the go and therefore it allows the simulation of larger systems, compared to earlier approaches [20].

The generalized coordinates of the systems are the local frame rotations (of frame $j$ with respect to frame $j - 1$). We further use a unit-quaternion representation of the rotations that has several advantages. First of all it avoids complications due to
multiple elementary rotations and singularities, secondly has the advantage that the unity of quaternions is a quadratic invariant, therefore easily preserved by existing quadratic-preserving methods.

There are many possible extension of the present work. An extension to prismatic joints as well as rotations with angle restrictions is quite straightforward, but was left out for simplicity sake. Further, we have observed that the complexity of the present approach is quadratic with the number of joints, but a thorough investigation of the complexity and optimization of the method has not been considered. In particular, it would be interesting to investigate whether it is possible to apply recursive methods to obtain $O(N)$ computational complexity, like in the current Newton–Euler formulation of mechanics [5]. In addition, the Lagrangian approach can be generalized to a discrete variational approach by doing an appropriate discretization of the Lagrangian and doing variation with respect to the discrete variables in the spirit of [14]. In this way, the discrete system inherits the underlying geometrical properties of the original system. This approach might be particularly interesting for long time simulations where it is important to conserve energy and/or momentum or other conserved quantities of the system, or in the context of flexible bodies and nonlinear beam theory.

REFERENCES

[1] E. Celledoni, A. Marthinsen and B. Owren, Commutator-free Lie group methods, Future Generations Computer Systems, 19 (2003), 341–352.
[2] L. Dieci, R. D. Russell and E. S. van Vleck, Unitary integrators and applications to continuous orthonormalization techniques, SIAM J. Num. Anal., 31 (1994), 261–281.
[3] F. Diele, L. Lopez and R. Peluso, The Cayley transform in the numerical solution of unitary differential systems, Adv. Comput. Math., 8 (1998), 317–334.
[4] E. Eich-Soellner and C. Führer, Numerical Methods in Multibody Dynamics, European Consortium for Mathematics in Industry. B. G. Teubner, Stuttgart, 1998, 290 pp.
[5] R. Featherstone, Rigid Body Dynamics Algorithms, Springer, New York, 2008.
[6] J. Flatlandsmo, T. Smith, O. Halvorsen and T. S. T. Impelluso Modeling stabilization of crane-induced ship motion with gyroscopic control using the moving frame method, Journal of Computational and Nonlinear Dynamics, 14 (2019).
[7] H. Goldstein, Classical Mechanics, Second edition, Addison-Wesley Series in Physics, Addison-Wesley Publishing Co., Reading, Mass., 1980.
[8] E. Hairer, C. Lubich and G. Wanner, Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations, Second edition, Springer Series in Computational Mathematics, 31. Springer-Verlag, Berlin, 2006.
[9] M. O. Hestevik, K. O. Austefjord, L.-K. S. Larsen and T. Impelluso, Modelling subsea ROV robotics using the moving frame method, International Journal of Dynamics and Control, 7 (2019), 1306–1320.
[10] T. J. Impelluso, The moving frame method in dynamics: Reforming a curriculum and assessment, International Journal of Mechanical Engineering Education, 6 (2018), 158–191.
[11] A. Iserles, M. P. Calvo and A. Zanna, Runge-Kutta methods for orthogonal and isospectral flows, Appl. Numer. Math., 22 (1996), 153–163.
[12] A. Iserles, H. Z. Munthe-Kaas, S. P. Nørsett and A. Zanna, Lie-group methods, Acta Numerica, Cambridge Univ. Press, Cambridge, 9 (2000), 215–365.
[13] L. O. Jay, Preserving Poisson structure and orthogonality in numerical integration of ordinary differential equations, Comput. Math. Appl., 48 (2004), 237–255.
[14] M. Leok, L. Taeyoung and N. H. McClamroch, Global Formulations of Lagrangian and Hamiltonian Dynamics on Manifolds. A Geometric Approach to Modeling and Analysis, Interaction of Mechanics and Mathematics, Springer, Cham, 2018.
[15] C. Lubich, U. Nowak, U. Pöhlle and C. Engstler, MEXX-Numerical Software for the Integration of Constrained Mechanical Multibody Systems, Technical Report Technical Report SC 92–12, ZIB Berlin, 1992.
[16] J. E. Marsden and T. S. Ratiu, *Introduction to Mechanics and Symmetry. A Basic Exposition of Classical Mechanical Systems*, Second edition, Texts in Applied Mathematics, 17. Springer-Verlag, New York, 1999.

[17] A. Müller, *Screw and Lie group theory in multibody kinematics*, Multibody Syst. Dyn., 43 (2018), 37–70.

[18] H. Murakami and T. J. Impelluso, *Moving Frame Method in Dynamics-A Geometrical Approach*, Pearson Publishing, 2019.

[19] A. A. Shabana, R. A. Wehage and Y. L. Hwang, *Projection methods in flexible multibody dynamics. Part II: Dynamics and recursive projection methods*, International Journal for Numerical Methods in Engineering, 35 (1992), 1941–1966.

[20] T. Rykkje, *Lie Groups and the Principle of Virtual Work Applied to Systems of Linked Rigid Bodies*, Master thesis, Universitetet i Bergen, 2018.

[21] J. M. Sanz-Serna and M. P. Calvo, *Numerical Hamiltonian Problems*, Applied Mathematics and Mathematical Computation, 7. Chapman & Hall, London, 1994.

[22] R. von Schwering, *MultiBody Systems Simulation. Numerical Methods, Algorithms, and Software*, Lecture Notes in Computational Science and Engineering, 7. Springer-Verlag, Berlin, 1999.

Received March 2019; revised September 2019.

E-mail address: nsatran@outlook.com
E-mail address: Antonella.Zanna@math.uib.no