University of Bergen
Department of Mathematics

Lagrangian dynamics for solid multi-body systems with the Moving Frame formalism

Author:
Nicolai Mikal Sætran

Supervisors:
Antonella Zanna Munthe-Kaas
Thomas J. Impelluso

UNIVERSITETET I BERGEN
Det matematisk-naturvitenskapelige fakultet

November, 2018

Contents

Abstract

Classical mechanics is the study of the motion of particles, solid bodies or of systems of bodies, and the effects of forces and moments on them. The dynamical behavior may either be studied by the application of Newtonian mechanics, Lagrangian mechanics or Hamiltonian Mechanics. Either approach produces mathematical equations describing the time evolution of the multi-body system. The resulting equations of motion for many bodies are typically non-linear and of large scale.

The Lagrangian approach is more abstract than the Newtonian approach in the sense that it employs more advanced theory from mathematics, and has therefore traditionally mostly been a topic studied in mathematics, physics or on master level engineering. The Moving Frame Method by H. Murakami et. Impelluso [MI19]. is a formalism which makes advanced results from classical mechanics and group theory available to bachelor level engineers. The method allows engineers without deep understanding of these fields to describe the dynamic behavior of systems that would otherwise be too complex to approach. The method is therefore a framework on which engineers may rely without having a background in group theory or calculus of variations.

In this thesis the focus will be on further generalizing the multi-body Moving Frame Method. We will restrict ourselves to multi-body systems with generalized coordinates which are all free rotations. These systems may all be idealized as N-body three-dimensional pendulums (These are sometimes named open kinematic chains). We seek to tackle in particular three challenges associated with the derivation and simulation of multi-body systems:

Coordinate-free description Most treatments of the dynamics (including the moving frame method) of solid bodies undergoing rotation employ Euler-angles to describe the rotational kinematics. The resulting equations of motion are typically long expressions of cosine and sine terms. For complex systems, these expressions 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 computing the solution. Another issue with the application of local coordinates is the presence of singularities in the equations of motion when studying three-degree-of-freedom rotations. These singularities originate from gimbal lock and produce significant difficulties. Inspired by the approach taken by Leok et.al. [TLM17], we seek to eliminate the gimal lock

by directly applying the representation space of the rotation groups in the formulation of the equations of motion under the Moving Frame Formalism. But where Leok et.al. limits the use of $SO(3)$ and $SE(3)$ to dynamics of single bodies, this thesis will apply these groups to multi-body systems.

Algorithmic generation of the equations of motion The large expressions in the equations of motion are due to the complicated kinematic expression that arises from long sequences of rotation matrices with sines and cosines in the entries. Instead of generating these expressions beforehand, we will derive expression for generating the matrix-terms of the equations of motion algorithmically during evaluation of the ODE-functions.

Quaternion representation The direct application of SO(3) as a representation space unfortunately causes some difficulties in practical computations. The $\mathbb{R}^{3\times3}$ rotation matrices take up more memory than strictly necessary, and correcting errors caused by rounding orthogonality is a costly venture. Therefore we have chosen to employ the more compact quaternion representation of orientations during computations. Quaternions are well known for the fact that monitoring and correcting errors on the quaternions is much simpler in comparison to the full representation for the rotation-matrices, and thus provide a attractive alternative representation. We will apply the unit quaternions to the equations of motion for a multi body system.

The structure of the thesis is as follows.

In *Chapter 1* the relevant properties of $SO(3)$, $SE(3)$ and the unit quaternions $\mathbb{S}^3$ from group theory are presented. Kinematics under the Moving Frame formalism is presented, and the kinematic relations for the N-body pendulum is derived.

In *Chapter 2* the main ideas of Lagrangian mechanics and Calculus of Variation is presented along with group theory results regarding variation of the $SE(3)$ group. Then finally the Euler-Lagrange equations are derived with the Moving Frame formalism. The resulting Euler-Lagrange equations are written in terms of the B and $\dot{B}$ matrices. The main contribution of this thesis is the derivation of the general expressions for these matrices, for long n-body three-dimensional pendulums.

In *Chapter 3* the equations of motion are rewritten in preparation for

numerical simulation, and suitable numerical schemes are suggested. A number of simulations are performed on a single body system, a 2-body system and and a 4-body system. Errors in the computed equations are analyzed to determine the accuracy and stability of the numerical solution. Finally toward the end a number of forced systems are presented. These are intended to demonstrate the ease with which external forces may be applied to the system.

Acknowledgements

I would like to thank Antonella Zanna Munthe-Kaas and Thomas J. Impelluso for valuable advice and help during the work on this thesis.

Notation

Abbreviation:

MFM:	Moving Frame Method
GL1, GL2, GL3:	Gauss-Legendre methods of order 1, 2 and 3

Groups:

$SO(3),\ SE(3), \mathbb{S}^3$	Sprecial orthogonal group, Special euclidean group and group of unit quaternions
$\mathfrak{so}(3), \mathfrak{se}(3), T_I\mathbb{S}^3$	Lie algebra of SO(3), SE(3) and the unit quaternions
$\hat{\omega}, \Omega, \xi$	Elements of $\in \mathfrak{so}(3), \mathfrak{se}(3), T_I\mathbb{S}^3$ respectively.

Other notational conventions:

- $\underline{\mathrm{x}}$ Time-dependent position vectors in $\mathbb{R}^3$
- $\underline{\mathrm{s}}$ Fixed position vectors in $\mathbb{R}^3$
- $\underline{\mathrm{v}}$ Translational velocity vectors in $\mathbb{R}^3$
- $\boldsymbol{y}$ Vector valued dependent ODE function variable.

MFM notation:

q	Essential generalized coordinates
$\dot{q}$	Essential generalized velocity vector
$\boldsymbol{\omega}$	Essential generalized velocity vector for 3D rotations
δq	Virtual essential generalized displacement
$\boldsymbol{\eta}$	Virtual essential generalized displacement for 3D rotations
X	System generalized coordinates
$\dot{X}$	System generalized velocity
δX	Virtual system generalized velocity
$R^{(j)}$	Absolute rotation matrix
$R^{(j/i)}$	Relative rotation matrix
$\boldsymbol{e}^{(j)}$	Rotating frame
$E^{(j)}$	Absolute frame connection matrix
$E^{(j/i)}$	Relative frame connection matrix
$\overline{e}^{(j)}$	Rotating and translating frame

Chapter 1

Preliminaries - Kinematics and matrix groups

Kinematics is concerned with the study of motion and its evolution in time without a consideration for the effects of forces and potentials. This thesis will deal with the motion of rigid bodies. As deformable bodies will not be considered, the term body will mean rigid body.

multi-body systems simultaneously undergo rotation and translation. The object of choice for representing the rotation of such systems are the rotation matrices. These matrices form the classical matrix group $SO(3)$, and in this section a short summary is given of the results from group theory that will used during the derivation of the system kinematics and equations of motion. Also used in the derivations is the group of euclidean motion $SE(3)$ which provides a description of systems simultaneously undergoing rotation and translation, and some results on this group will be summarized.

The Moving Frame Method - shortened MFM - is a formalism which applies the concept of the moving frame developed by Elie Cartan and a notation proposed by Thèodore Frenkel. The method places local frames of reference to the body in question, and study the motion as linear transformations of the moving frames. A summary of kinematics under the Moving Frame formalism will be given, in order to introduce the notation used by the method.

A summary of the the group of unit quaternions will be given, with particular focus on its real-matrix representations. The group of unit quaternions are useful as they are homeomorphic to $SO(3)$, and thus provides a alternative for the representation of rotations.

1.1 Lie Matrix Groups

The classical groups $SO(3)$ and $SE(3)$ are subsets of the general linear group $GL(n)$. Matrix group theory provides many important results necessary for developing the ordinary differential equations for solid body systems, and in this section some results regarding the classical rotation groups will be summarized. The summary will be restricted to the results required for the development of the dynamics.

1.1.1 Lie matrix groups

A group is a set G and a operation $\circ$. Given two arbitrary elements of the group:

$$g_1, g_2, I \in G$$

They must satisfy the group axioms:

Closure $g_1 \circ g_2 \in G$

Unity $g \circ I = I \circ g = g$

Inverse $g \circ g^{-1} = g^{-1} \circ g = I$

Lie matrix groups are groups of linear transformations that satisfies the group axioms. The group operation $\circ$ will be the matrix product throughout this thesis.

Lie Algebra The tangent space at the identity is the Lie algebra of the group denoted $\mathfrak{g}$. The Lie algebra may be defined as the tangent vectors of smooth paths through the identity. Let $g(t) \in G$ with $g(0) = I$ be smooth paths in G, then $\mathfrak{g}$ is $\dot{g}(0)$. The Lie algebra is a linear space closed under the Lie bracket. Let $\zeta_1, \zeta_2 \in \mathfrak{g}$, then $[\zeta_1, \zeta_2] \in \mathfrak{g}$.

The exponential function The tangent space and the group is associated through the exponential function which maps the tangent space onto the group $\mathrm{e}^\bullet : \mathfrak{g} \to G$. Given a Lie Algebra element ξ. For matrix groups the exponential function $\mathrm{e} : \mathfrak{g} \to GL(\mathbb{R}, n)$ defined as.

$$\mathrm{e}^\zeta = \sum_{k=0}^{\infty} \frac{\zeta^k}{k!}, \qquad \mathrm{e}^\zeta \in G,\ \zeta \in \mathfrak{g}$$

Adjoint The adjoint is a linear transformation that maps tangents from one tangent space on the group to another. Let $g \in G$ and $\sigma(t) = I + \zeta t + \mathcal{O}(t^2)$ be a smooth curve on G. The right adjoint representation si defined as:

$$Ad_g^r(\zeta) = \tfrac{\mathrm{d}}{\mathrm{d}t} g^{-1}\sigma(t) \in \mathfrak{g}\Big|_{t=0} = g^{-1}\zeta g$$

There is also the right-adjoint we also have the left adjoint $Ad_g^l(\xi) = g\xi g^{-1}$. Only the right adjoint will be used. So for simplicity of notation, $Ad_g(\bullet)$ will always be the right adjoint throughout the thesis.

In the context of matrix groups, the adjoint can be interpreted as a operator. It is linear in the argument and respects the commutator.

$$\begin{aligned} Ad_{g_1} \circ Ad_{g_2} &= Ad_{g_1 g_2} \\ Ad_g(\xi_1 + \xi_2) &= Ad_g(\xi_1) + Ad_g(\xi_2) \\ Ad_g([\xi_1, \xi_2]) &= \big[Ad_g(\xi_1), Ad_g(\xi_2)\big] \end{aligned}$$

Because of the linearity of the adjoint, it can be expressed as a matrix vector product. The linear relation for a arbitrary invertible matrix $A \in \mathbb{R}^{n\times n}$ and a matrix $X \in \mathbb{R}^{n\times n}$ is:

$$Ad_A X = A^{-1} X A = C$$

Where the vectorized representation of the matrix C can be computed by the matrix vector product.

$$vec(C) = (A^T \otimes A^{-1}) vec(X)$$

(see [HJ85, Chapter 4, p. 254])

The derivative of the Adjoint The little adjoint is the derivative of the adjoint operator at the identity.

For $\zeta, \nu \in \mathfrak{g}$ and a curve $\sigma(t) = I + \nu t + \mathcal{O}(t^2) \in G$ the right-adjoint is.

$$\begin{aligned} ad_\nu^r(\zeta) &= \tfrac{\mathrm{d}}{\mathrm{d}t} Ad_{\sigma(t)}\zeta|_{t=0} \\ &= [\zeta, \nu] \end{aligned}$$

Or written as an operator:

$$ad_\nu^r(\bullet) = [\bullet, \nu]$$

Alternatively the left adjoint is the operator $ad_\nu^l(\bullet) = [\nu, \bullet]$. In the thesis, only the right adjoint will be used, so therefore $ad_\nu(\bullet)$ will always denote the right adjoint.

There is an important identity relating the adjoint operator to the exponential of the little adjoint. See [IMKNZ00, p. 237].

$$\begin{aligned} Ad_{\mathrm{e}^{t\zeta}}\nu &= \mathrm{e}^{tad(\zeta)}\nu \\ &= \sum_{k=1}^{\infty} \frac{t^k}{k!} ad_\zeta^k \nu \end{aligned} \tag{1.1}$$

1.2 SO(3)

The special orthogonal group $SO(3)$ is the set of isometric linear transformations of $\mathbb{R}^3$ with determinant 1.

Definition 1.2.1 (SO(3))**.** The group of orthogonal matrices with determinant 1.

$$SO(3) = \left\{ R \in \mathbb{R}^{3\times 3} : R^T R = R R^T = I, det(R) = +1 \right\}$$

For each group element

$$R^{-1} = R^\top, \quad R \in SO(3)$$

More intuitively, $SO(3)$ is the group of rotation matrices in the three-dimensional euclidean space.

1.2.1 The Lie algebra of SO(3)

The Lie algebra of $SO(3)$ are the skew symmetric matrices.

$$\hat{\omega} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \in \mathfrak{so}(3) \tag{1.2}$$

Where the hat notation denote the skew-symmetry of the matrix. The tangent at R is:

$$\dot{R} = R\hat{\omega} \tag{1.3}$$

is the tangent at R.

The elements of the Lie algebra can be identified with a three-term angular velocity vector. The associated angular velocity vector is.

$$\underline{\omega} = \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix}$$

This vector can be interpreted as a angular velocity $\dot{\theta} = |\underline{\omega}|$ about the unit vector $\underline{n} = \frac{\underline{\omega}}{|\underline{\omega}|}$.

1.2.2 Exponential Function

The exponential function on $SO(3)$ $\mathrm{e}^{t\hat{\omega}} : \mathfrak{so}(3) \to SO(3)$ is:

$$\mathrm{e}^{\hat{\omega}} = \sum_{k=0}^{\infty} \frac{t^k}{k!} \hat{\omega}^k$$

Example 1.2.1 (Fundamental rotation matrices)**.** The exponential function applied to the one-parameter unit generators for SO(3) gives the one-parameter subgroups of SO(3). A rotation is generated by a angular velocity about the first axis by the exponential function as follows:

$$\mathrm{e}^{(\theta \hat{e}_1)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta(t)) & -\sin(\theta(t)) \\ 0 & \sin(\theta(t)) & \cos(\theta(t)) \end{bmatrix} = R_1$$

Which is the rotation matrix, rotating $\mathbb{R}^3$ about $\boldsymbol{e}_1$.

1.2.3 Adjoint Operator

The adjoint operator for the SO(3) group, maps elements from the Lie algebra onto itself:

$$Ad_R \hat{\omega} = R^\top \hat{\omega} R \quad R \in SO(3),\ \hat{\omega} \in \mathfrak{so}(t) \tag{1.4}$$

As mentioned in 1.1 the adjoint operator is linear, and can be rewritten as a matrix-vector product

$$Ad_R \hat{\omega} = C, \quad vec(C) = \left(R^\top \otimes R^\top \right) \mathrm{vec}(\hat{\omega})$$

As the adjoint maps $\mathfrak{so}(3)$ onto itself $Ad_R : \mathfrak{so}(3) \to \mathfrak{so}(3)$, there must exist a matrix $N \in \mathbb{R}^{3\times 3}$ such that.

$$C\underline{\omega} = vec(\widehat{N\underline{\omega}})$$

And therefore an expression for the Adjoint is:

$$Ad_R(\hat{\omega}) = \widehat{N\underline{\omega}}$$

The matrix N has a very simple connection to R. In fact they are related by $N = R^\top$. We may see this by applying the identity 1.1. Now the right-adjoint operator can be rewritten (See Appendix).

$$Ad_{e^{\hat{\xi}t}}\hat{\omega} = e^{ad_{t\hat{\xi}}}\hat{\omega}, \quad \hat{\xi} \in \mathfrak{so}(3)$$

$$= \widehat{e^{-\hat{\xi}t}\hat{\omega}}$$

$$= \widehat{R^\top(t)\omega}$$

Yielding a simple and useful expression for the adjoint requiring no intermediate computations.

$$Ad_R(\hat{\omega}) = \widehat{R^\top(t)\hat{\omega}} \tag{1.5}$$

1.2.4 Kinematics in the Moving Frame Method - Rotations

The Moving Frame Method in [MI19] applies a specialized notation for denoting frames of reference. A short summary will be given in this section: Given two orthogonal frames of reference, one chosen as a fixed inertial frame and one as a body-attached moving frame rotated away from the inertial by the $SO(3)$ rotation matrix $R^{(j)}$. The inertial frame is defined as the standard basis with unit vectors $(\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3)$. Written with a shorthand notation we denote the inertial frame $\boldsymbol{e}^I$. The moving frame is a orthogonal basis denoted $\boldsymbol{e}^{(1)}$. And the relation between the two frames is:

$$\boldsymbol{e}^{(1)} = \boldsymbol{e}^I R^{(1)} \tag{1.6}$$

The addition of multiple frames follows the same definition. A second frame $\boldsymbol{e}^{(2)}$ is expressed relative to the first by the relative rotation matrix:

$$\boldsymbol{e}^{(2)} = \boldsymbol{e}^{(1)} R^{(2/1)}$$

And a j'th frame $\boldsymbol{e}^{(j)}$ is expressed in the parent frame $\boldsymbol{e}^{(j/j-1)}$ by the relative rotation matrix:

$$\boldsymbol{e}^{(j)} = \boldsymbol{e}^{(j-1)} R^{(j/j-1)}$$

Vectors and points are then specified by frame, and can be transformed between frames with 1.6.

$$\boldsymbol{e}^{(j)}\underline{r}(t) = \boldsymbol{e}^{(j-1)} R^{(j/j-1)}\underline{x}, \quad \underline{x} \in \mathbb{R}^3$$

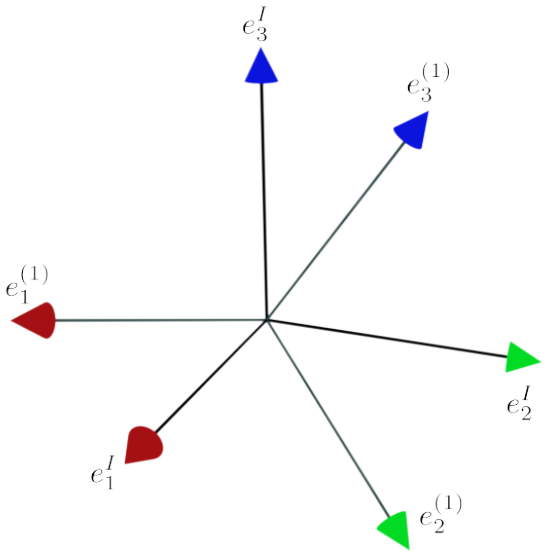


Figure 1.1: Inertial frame e^I and moving frame $e^{(1)}$

Moving vectors in the opposite direction is performed by a transformation with the inverse $SO(3)$, which is simply the transposed matrix.

$$e^I = e^{(j)} {R^{(j)}}^\top \tag{1.7}$$

Ths for a vector we may

$$\underline{r}(t) = \boldsymbol{e}^I \underline{x}^{(I)}(t) = \boldsymbol{e}^I R^{(j)} {R^{(j)}}^\top \underline{x}(t) = \boldsymbol{e}^{(j)} {R^{(j)}}^\top \underline{x}(t)$$

Velocity The instantaneous time-rate of change of orientation for the moving frame is the time derivative of the rotation matrix relating the two frames.

$$\dot{\boldsymbol{e}}^{(1)} = \boldsymbol{e}^I \dot{R}^{(1)}$$

Inserting 1.6 we recognize from 1.3 that the angular velocity of the moving frame, expressed in the frame itself is an element of the Lie algebra.

$$\dot{\boldsymbol{e}}^{(1)} = \boldsymbol{e}^{(1)}\hat{\omega}^{(1)}$$

With this notation, the velocity of a point in the moving frame is interpreted as:

$$\frac{\mathrm{d}}{\mathrm{d}t}x^{(1)} = \frac{\mathrm{d}}{\mathrm{d}t}\left(\boldsymbol{e}^{(1)}\underline{x}(t)\right) = \boldsymbol{e}^{(1)}(t)\hat{\omega}^{(1)}\underline{x}(t) + \boldsymbol{e}^{(1)}(t)\dot{\underline{x}}(t) \tag{1.8}$$

The angular velocity of a second frame, expressed relative to the first frame is the sum of the relative angular velocity and the angular velocity of the parent frame transformed into the local frame.

$$\begin{aligned}\tfrac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{e}^{(2)}(t) &= \boldsymbol{e}^{(2)}\left({R^{(2/1)}}^{\top}\hat{\omega}^{(1)}R^{(2/1)} + \hat{\omega}^{(2/1)}\right)\\ &= \boldsymbol{e}^{(2)}\left(Ad_{R^{(2/1)}}(\hat{\omega}^{(1)}) + \hat{\omega}^{(2/1)}\right)\end{aligned}$$

In general for k'th frame:

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{e}^{(k)}\underline{x}(t) = \boldsymbol{e}^{I}\frac{\mathrm{d}}{\mathrm{d}t}R^{(k)}(t)\underline{x}(t) + \boldsymbol{e}^{(k)}\dot{\underline{x}}(t)$$

Under this formalism, the Lie algebra has a more intuitive interpretation as the angular velocity of the moving frame expressed in the local frame itself.

1.2.5 Rewriting the adjoint

When deriving the rotational velocity we encounter:

$$\frac{\mathrm{d}}{\mathrm{d}t}R^{(k)}(t)$$

For application later in the thesis we wish to rewrite the expression for the derivative of the k'th rotation matrix. This sum can be rewritten as follows:

$$\begin{aligned}\tfrac{\mathrm{d}}{\mathrm{d}t}R^{(k)}(t) = &\left(\tfrac{\mathrm{d}}{\mathrm{d}t}R^{(1)}\right)R^{(k/1)} + \cdots + R^{(j-1)}\left(\tfrac{\mathrm{d}}{\mathrm{d}t}R^{(j/j-1)}\right)R^{(k/j-1)} + \cdots\\ &\cdots + R^{(k-1)}\left(\tfrac{\mathrm{d}}{\mathrm{d}t}R^{(k/k-1)}\right)\\ = &R^{(1)}\hat{\omega}^{(1)}R^{(k/1)} + \cdots + R^{(j)}\hat{\omega}^{(j/j-1)}R^{(k/j-1)} + \cdots\\ &\cdots + R^{(k)}\hat{\omega}^{(k/k-1)}\\ = &R^{(k)}\Big({R^{(k/1)}}^{\top}\hat{\omega}^{(1)}R^{(k/1)} + \cdots + {R^{(k/j)}}^{\top}\hat{\omega}^{(j/j-1)}R^{(k/j)} + \cdots\\ &\cdots + \hat{\omega}^{(k/k-1)}\Big)\end{aligned}$$

We recognize the adjoint operator 1.4 in the expression. Applying this, we see that the absolute velocity of frame k is the sum of the relative velocities of the preceding relative frames in the tangent space of $R^{(k)}$.

$$\begin{aligned}\tfrac{\mathrm{d}}{\mathrm{d}t}R(t) =& R^{(k)}\left(Ad_{R^{(k/1)}}(\hat{\omega}^{(1)}) + \cdots + Ad_{R^{(k/j)}}(\hat{\omega}^{(j/j-1)}) + \cdots + \hat{\omega}^{(k/k-1)}\right) \\ =& R^{(k)}\left(\sum_{j=1}^{k} Ad_{R^{(k/j)}}\left(\hat{\omega}^{(j/j-1)}\right)\right) \end{aligned} \tag{1.9}$$

This sum of expressions can be rewritten to matrix-vector form by applying eq.(1.5) that was previously obtained. Thus the derivative is rewritten as:

$$\begin{aligned}\tfrac{\mathrm{d}}{\mathrm{d}t}R^{(k)} &= R^{(k)}\left(\sum_{j=1}^{k} Ad_{R^{(k/j)}}\left(\hat{\omega}^{(j/j-1)}\right)\right) \\ &= R^{(k)}\left(\sum_{j=1}^{k} \overbrace{R^{(k/j)^{\top}}\underline{\omega}^{(j/j-1)}}\right) \end{aligned} \tag{1.10}$$

Note on notation

The SO(3) group will be used to represent configurations of solid bodies in three dimensions. The conventions in this thesis will largely follow that of the moving frame method by Murakami et.al. [MI19] The orientation of frame j relative to the inertial frame is denoted $R^{(j)}$. A configuration expressed relative to another moving frame is expressed by relative rotation matrices. The rotation matrix between frames k and j will be expressed as $R^{(k/j)}$. In the instances where sums are used, the following convention is applied: $R^{(k/0)} = R^{(k)}$ and $R^{(n/n)} = I$. Relative angular velocities between frames k and j will be denoted $\hat{\omega}^{(k/j)}$. In instances where using sums, the following convention is applied: $\hat{\omega}^{(k/0)} = \hat{\omega}^{(k)}$ and $\hat{\omega}^{(j/j)} = 0$.

1.3 Euclidean motion - SE(3)

Multi-body systems undergo both translation and rotation, which are represented as rigid transformations and may be named Euclidean motion. The $SE(3)$ group is described by the pair $(R, \underline{x})$ and represent a combined rotation and translation.

$$SE(3) = \left\{ (R, \underline{x}) \in \mathbb{R}^{3\times 3} \times \mathbb{R}^3 : R \in SO(3) \right\}$$

The group composition is the semi-direct product of $SO(3)$ and $\mathbb{R}^3$, which can be represented as a linear transformation with matrix representation:

$$E = \begin{bmatrix} R & \underline{x} \\ 0 & 1 \end{bmatrix}, \quad E \in SE(3)$$

This matrix can be interpreted as representations of $SE(3)$. The Moving Frame formalism makes use of both representations of $SE(3)$.

With the matrix representation, $SE(3)$ is a group with the matrix product as group composition. The identity of the group is the identity matrix $I \in \mathbb{R}^3$ and the group inverse is:

$$E^{-1} = \begin{bmatrix} R^T & -R^T\underline{x} \\ 0 & 1 \end{bmatrix}$$

1.3.1 Lie algebra

The Lie algebra of SE(3) is defined as:

$$\mathfrak{se}(3) = \left\{ \begin{bmatrix} \hat{\omega} & \underline{v} \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{3\times 3} : \hat{\omega}, \underline{v} \in \mathbb{R}^3 \right\}$$

The tangent space of $SE(3)$ is given by

$$\dot{E} = E\Omega$$

And can be understood as the angular and translational velocity of the object undergoing euclidean motion.

1.3.2 Exponential function

The exponential function $\mathrm{e} : \mathfrak{se}(3) \to SE(3)$ is defined as:

$$\mathrm{e}^{\Omega t} = \sum_{j=0}^{\infty} \frac{t^k}{k!} \Omega^k$$

1.3.3 Kinematics with SE(3)

The $SE(3)$ group may be taken to represent orientation and position for moving frames. For a rotating and translating frame in euclidean space as displayed in 1.2 We take

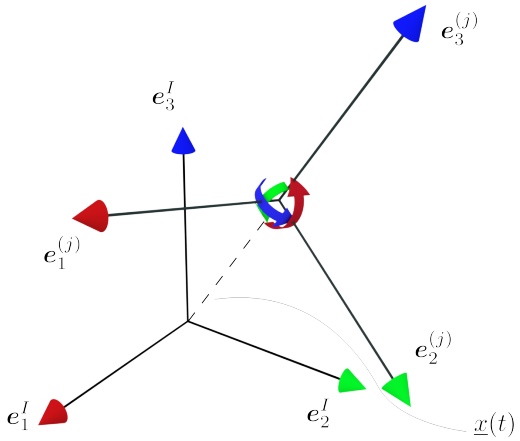


Figure 1.2: Inertial and rotating/translating frame

$$(\boldsymbol{e}^{(j)}, \mathbf{0})$$

to represent the inertial frame. Shorthand notation for the frame is

$$\overline{\boldsymbol{e}}^I = (\boldsymbol{e}^{(j)}, \mathbf{0})$$

To describe the kinematics of the moving frame, the position and orientation of the moving frame is represented by the euclidean group $SE(3)$ and is denoted

$$\overline{\boldsymbol{e}}^{(1)} = \left(R^{(1)}, \underline{x}^{(1)}\right)$$

And the relation between the frame and the inertial is expressed by the linear transformation form of the $SE(3)$ group.

$$\overline{\boldsymbol{e}}^{(1)} = \overline{\boldsymbol{e}}^I E^{(1)}, \quad E^{(1)} \in SE(3)$$

Thus we may understand $SE(3)$ as a frame, rotated and translated from the inertial frame.

Given a second frame expressed relative to the first, the absolute expression for the second frame is:

$$\overline{\boldsymbol{e}}^{(2)} = \overline{\boldsymbol{e}}^{(1)} E^{(2/1)}$$

Velocity With the inertial frame taken to be stationary, the velocity of the first frame is a element of the Lie algebra:

$$\dot{\overline{\boldsymbol{e}}}^{(1)} = \overline{\boldsymbol{e}}^{(1)}\Omega^{(1)}$$

The velocity of the second frame is the sum of the relative velocity and the tangent of the parent frame transformed into the tangent space of the second frame.

$$\dot{\overline{\boldsymbol{e}}}^{(2)} = \overline{\boldsymbol{e}}^{(2)}(E^{(2/1)^{-1}}\Omega^{(1)}E^{(2/1)} + \Omega^{(2/1)})$$

And the time-rate of change for arbitrary frame k is:

$$\overline{\boldsymbol{e}}^{(k)} = \overline{\boldsymbol{e}}^{(k)}\Omega^{(k)},$$

where again the tangent of the relative frame k, is the sum of the tangent of the previous parent frame transformed by the adjoint transformation to the local frame and the relative velocity.

$$\Omega^{(k)} = E^{(k/k-1)^{-1}}\Omega^{(k/k-1)}E^{(k/k-1)} + \Omega^{(k/k-1)}$$

1.4 Kinematics of the n-body pendulum

In this section, the kinematic of the multi-body system idealized as a n-body pendulum will be derived. The derivation will be done in terms of $SE(3)$ matrices under the Moving Frame formalism.

The systems under consideration are solid bodies interconnected at three-dimensional joints with the first link fixed to the origin. These systems can be thought to form a N-body three-dimensional pendulum. To express the motion of the bodies in the system, local coordinate frames represented by relative $SE(3)$ matrices are assigned to the center of mass of the bodies, and aligned with the principle axes of the moment of inertia tensor. Thus the rotation and translation of each body in the system is expressed as a element of the special euclidean group, and the angular and translational velocity are elements of its tangent space. A illustration with a length of 4 is displayed in 1.3, where the three arrows around the center of mass signifies that the body is free to rotate about every axis. The following notation convention is

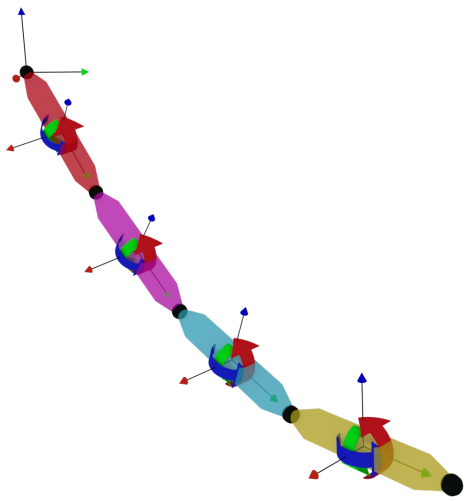


Figure 1.3: 4-body three-dimensional pendulum

established: Fixed distances between joints and centers of mass are denoted $\underline{s}_{i,j}$ while the time-dependent position is expressed $\underline{x}(t)$. The first of the subscript on $\underline{s}_{i,j}$ refers to link number counting from origo, while the second indicates whether the vector describes distance from joint to center of mass or distance from center of mass (CM) to next joint, counting from origo and outward along the chain. This convention is displayed in 1.4.

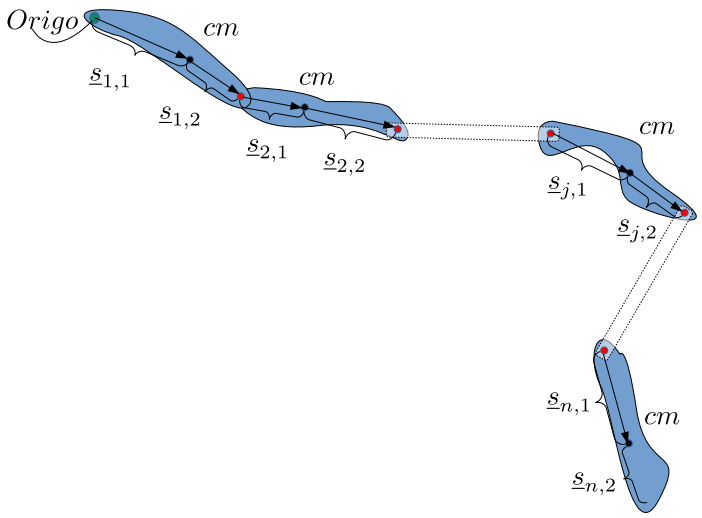


Figure 1.4: Labeling convention for distances between joints and centers of mass (CM)

Kinematics The first frame is the product of a rotation and a translation.

$$\begin{aligned}\overline{\boldsymbol{e}}^{(1)} &= \overline{\boldsymbol{e}}^{I} E^{(1)} \\ &= \overline{\boldsymbol{e}}^{I} \begin{bmatrix} R^{(1)} & \underline{0} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I & \underline{s}_{1,1} \\ 0 & 1 \end{bmatrix} \\ &= \overline{\boldsymbol{e}}^{I} \begin{bmatrix} R^{(1)} & R^{(1)}\underline{s}_{1,1} \\ 0 & 1 \end{bmatrix}\end{aligned}$$

Thus, orientation and position of the first body is.

$$\overline{\boldsymbol{e}}^{(1)} = \overline{\boldsymbol{e}}^{I}(R^{(1)}, R^{(1)}\underline{s}_{1,1}) \tag{1.11}$$

The the relative reference between arbitrary bodies $j+1$ and j are composed from the product of 3 $SE(3)$ matrices and denoted as follows:

$$\begin{aligned}E^{(j/j-1)} &= \begin{bmatrix} I & \underline{s}_{j-1,2} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R^{(j/j-1)} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I & \underline{s}_{j,1} \\ 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} R^{(j/j-1)} & \underline{s}_{j-1,2} + R^{(j/j-1)}\underline{s}_{j,1} \\ 0 & 1 \end{bmatrix}\end{aligned}$$

The absolute reference frame is then:

$$\begin{aligned}\overline{\boldsymbol{e}}^{(j)} &= \overline{\boldsymbol{e}}^{(j-1)} E^{(j/j-1)} \\ &= \begin{bmatrix} R^{(j)} & x_{j-1}(t) + R^{(j-1)}(\underline{s}_{j-1,2} + R^{(j/j-1)}\underline{s}_{j,1}) \\ 0 & 1 \end{bmatrix}\end{aligned}$$

Thus the orientation and position of frame j is given by:

$$\overline{\boldsymbol{e}}^{(j)} = (R^{(j)}, \underline{x}_{j-1}(t) + R^{(j-1)}(\underline{s}_{j,2} + R^{(j/j-1)}\underline{s}_{j,1})) \tag{1.12}$$

The velocity and acceleration are easily computed from the definitions. However, these expressions will not be required in the derivation of the Euler-Lagrange equations, and are therefore left out.

1.5 The $\mathbb{S}^3$ of unit Quaternions

The unit quaternions is the three-sphere $\mathbb{S}^3$ in $\mathbb{R}^4$ under quaternion multiplication. The group of unit quaternions is well known for being homomorphic to SO(3), and thus can serve as an alternative description of rotation. A summary of quaternion algebra and their application to rotations is given, followed by a summary of the group properties and finally the real-matrix representation of quaternions.

1.5.1 Definitions

Quaternions extends complex numbers to four dimensional complex numbers. The set of all quaternions is denoted $\mathbb{H}$, and are the four-tuple in $\mathbb{R}^4$ with the quaternion-product as product operation. Let $q \in \mathbb{H}$.

$$q = q_0\mathbf{h} + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k}$$

Or written as a vector

$$q = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \in \mathbb{H},$$

with a shorthand notation:

$$q = \begin{bmatrix} q_0 \\ q_v \end{bmatrix}$$

Addition for quaternions have the same properties as vectors, while multiplication of two quaternions $q \star p,\ q, p \in \mathbb{H}$ is defined according to the following rules for multiplication of the basic elements:

$$\mathbf{h} \star \mathbf{h} = \mathbf{h} = 1, \mathbf{h} \star \mathbf{i} = \mathbf{i}, \mathbf{h} \star \mathbf{j} = \mathbf{j}, \mathbf{h} \star \mathbf{k} = \mathbf{k}$$
$$\mathbf{i} \star \mathbf{i} = \mathbf{j} \star \mathbf{j} = \mathbf{k} \star \mathbf{k} = -1$$
$$\mathbf{i} \star \mathbf{j} = \mathbf{k} = -\mathbf{j} \star \mathbf{i}, \mathbf{j} \star \mathbf{k} = \mathbf{i} = -\mathbf{k} \star \mathbf{j}, \mathbf{k} \star \mathbf{i} = \mathbf{j} = -\mathbf{i} \star \mathbf{k}$$

With the shorthand notation defined above, the quaternion product can be written:

$$p \star q = \begin{bmatrix} p_0 q_0 - p_v^\top q_v \\ p_0 q_v + q_0 p_v + \hat{p_v} q_v \end{bmatrix}$$

The complex conjugate is defined:

$$q^\dagger = \begin{bmatrix} q_0 \\ -q_v \end{bmatrix}$$

With which the norm is defined as:

$$|q| = \sqrt{q \star q^\dagger} = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}$$

The inverse is defined with the quaternion conjugate as.

$$q^{-1} = \frac{q^\dagger}{||q||}$$

Note on notation

In mechanics, the main interest is directed towards the quaternions of unit length ($|u| = 1$). The letter q is widely used to denote generalized coordinates in classical mechanics. Therefore the letter u will be used to denote the unit quaternions.

1.5.2 Rotations

Given the point coordinate $\underline{x} \in \mathbb{R}^n$, we define quaternion:

$$v = \begin{bmatrix} 0 \\ \underline{x} \end{bmatrix} \tag{1.13}$$

The unit quaternion defined as:

$$u = \begin{bmatrix} \cos\frac{1}{2}\theta \\ \sin\frac{1}{2}\theta \underline{n} \end{bmatrix}$$

where $\underline{n}$ is a unit-vector $\underline{n} \in \mathbb{R}^3$.

The conjugation:

$$\phi_u(v) = p \star v \star p^\dagger \tag{1.14}$$

rotates the complex part of v ($\underline{x}$), in the plane orthogonal to $\underline{n}$.

The antipodal pair u and $-u$ induces the a rotation with 1.14 of the imaginary component of the quaternion v. Thus the unit quaternions represent rotations i $\mathbb{R}^3$.

1.5.3 The group of unit quaternions $\mathbb{S}^3$

The unit quaternions form a group [Sti08].

The set of all quaternions of unit length,

$$\mathbb{S}^3 = \left\{ (u_0, \underline{u}_v) \in \mathbb{R} \times \mathbb{R}^3 : u_0^2 + ||\underline{u}_v||^2 = 1 \right\}, \quad u \in \mathbb{H}$$

is a group with the quaternion product as group operation. Given $u, v \in \mathbb{S}^3$, they satisfy:

Closure $u \star v \in \mathbb{S}^3$

Identity $u \star I = I \star u = u, \quad I = \begin{bmatrix} 1 \\ \underline{0} \end{bmatrix}$

Inverse $u \star u^{-1} = u^{-1} \star u = I, \quad u^{-1} = \frac{u^\dagger}{||u||} = u^\dagger$

The group $\mathbb{S}^3$ is thus a group of rotations, where the pair of quaternions $\pm u$ rotates a pure complex quaternion through the conjugation 1.14

The Lie algebra of the unit quaternions are the pure imaginary quaternions.

$$T_I \mathbb{S}^3 = \left\{ \xi = \mathbb{S}^3 : \xi_0 = 0, \underline{\xi}_v = \mathbb{R}^3 \right\}$$

The elements of Lie algebra can be written:

$$\xi = u^\dagger \star \dot{u} \tag{1.15}$$

1.5.4 Quaternions as real matrices

Computations with quaternions may be represented as operations with real matrices. The derivation in the following section is largely taken from of [Lac07, Chapter 13]. First we observe that the product $u \star v$ is a linear transformation and can therefore be rewritten as a matrix-vector product $Q(u)v$, where

$$Q(u) = \begin{bmatrix} u_0 & -u_1 & -u_2 & -u_3 \\ u_1 & u_0 & -u_3 & u_2 \\ u_2 & u_3 & u_0 & -u_1 \\ u_3 & -u_2 & u_1 & u_0 \end{bmatrix}, \quad Q(u) \in \mathbb{R}^4 \tag{1.16}$$

Or by defining

$$u = \begin{bmatrix} u_0 \\ \underline{u}_v \end{bmatrix}, \quad \underline{u}_v = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}$$

The matrix form of the quaternion product can be written more compactly:

$$\begin{aligned} u \star v &= Q(u)v \\ &= \left[\begin{array}{c|c} u_0 & -u_v^T \\ \hline u_v & u_0 I + \hat{u}_v \end{array}\right] \left[\begin{array}{c} v_0 \\ \hline v_v \end{array}\right] \end{aligned}$$

In the same manner, the product $v \star u$ can be written as a linear transformation in p.

$$\begin{aligned} u \star v &= P(v)u \\ &= \left[\begin{array}{c|c} v_0 & -v_v^T \\ \hline v_v & v_0 I - \hat{v}_v \end{array}\right] \end{aligned} \tag{1.17}$$

The product of a complex conjugate quaternion is simply:

$$\begin{aligned} u^\dagger \star v &= Q(u^\dagger)v = Q^\top(u)v \\ u \star v^\dagger &= P(v^\dagger)u = P^\top(v)u \end{aligned}$$

Real matrix representation of quaternion rotations

With these definitions the conjugation map 1.14 is rewritten as a matrix vector product

$$\begin{aligned} \phi_v(u) &= Q(v)P^\top(v)u \\ &= \left[\begin{array}{c|c} 1 & \underline{0}^T \\ \hline \underline{0} & R(v) \end{array}\right] u \end{aligned} \tag{1.18}$$

where

$$R(u) = I_3 + 2u_0\hat{u}_v + 2\hat{u}_v^2 \tag{1.19}$$

The map $R(u)$ is a 2-to-1 homomorphism from $\mathbb{S}^3$ to $SO(3)$, $R(q) : \mathbb{S}^3 \to SO(3)$ mapping the antipodal quaternion pairs $\{\pm u\}$ to a single element on $SO(3)$.

Example 1.5.1 (Fundamental rotations)**.** Define the rotation quaternion about the $\underline{e}_i$ unit vector.

$$q = \begin{bmatrix} \cos(\theta/2) \\ \sin(\theta/2)\underline{e}_i \end{bmatrix}$$

Applying $R(u)$ to it and working out the terms returns the elementary rotation matrices about the $\underline{e}_i$ axes.

Angular velocity

We proceed with deriving a expression for the time-derivative of the rotation map in terms of the unit-quaternion Lie algebra. Renaming the rotation map $Q(\underline{u})P^\top(\underline{u}) = A(\underline{u})$ we take the time derivative.

$$\dot{A}(q) = \dot{Q}P^T + Q\dot{P}^T$$

and rewrite the expression

$$\begin{aligned} \dot{A}(q) &= P^T\dot{Q} + Q\dot{P}^T \\ &= P^TQQ^T\dot{Q} + QP^TP\dot{P}^T \\ &= \underbrace{QP^T}_{A}Q^T\dot{Q} + \underbrace{P^TQ}_{A}P\dot{P}^T \\ &= A\left(Q^T\dot{Q} + P\dot{P}^T\right) \end{aligned}$$

Where from the first to the second lines, the matrices are interchanged. This is because the matrices Q and P commute [Lac07, Chapter 13, p. 227]

Next we notice that the matrix expression can with 1.18 and 1.17 be rewritten as:

$$\begin{aligned} Q^T\dot{Q} &= Q(u^\dagger \star \dot{u}) \\ P\dot{P}^T &= P^T(u^\dagger \star \dot{u}) \end{aligned}$$

Using this together with the definition of the tangent 1.15 $\xi = u^\dagger \star \dot{u}$ we can rewrite:

$$\dot{A}(u) = A(u)(Q(\xi) + P^T(\xi))$$

It is easily seen that:

$$\dot{A} = A\left[\begin{array}{c|c} 0 & 0 \\ \hline \underline{0} & 2\hat{\xi}_v \end{array}\right]$$

Extracting the lower left block matrices.

$$\dot{R} = R(u)(2\hat{\xi}_v) = R(u)\hat{\omega} \tag{1.20}$$

From which we obtain the relation between $T\mathbb{S}^3$ and $\mathfrak{so}(3)$:

$$\omega = 2\xi_v, \quad \omega \in \mathfrak{so}(3),\ \xi \in T\mathbb{S}^3. \tag{1.21}$$

The angular velocity expressed as the vector representation of$\mathfrak{so}(3)$ are two times the length of the vector expressed in$T\mathbb{S}^3$.

Chapter 2

Rigid body dynamics

In context of classical mechanics, dynamics is the study of the effects of forces and potentials on a single or multiple solid bodies. Once a kinematic description of a system has been formulated, Newtonian- or Lagrangian mechanics provides the relation between forces and their effect on the system.

The kinematics of a multi-body system is determined by the constraints that it must obey. Within Lagrangian mechanics, these constraints may be imposed in two ways. The first is with a extra set of equations, which are then included with Lagrangian multipliers. The other approach, applied in this thesis, is the inclusion of the constraints by the introduction of generalized coordinates, which characterize the possible motion of the system. With a suitable choice of generalized coordinates, constraint forces can be ignored as we only study the effects of forces which agree with the chosen coordinates.

The kinematic description in terms of rotation matrices was written in anticipation of the Lagrangian formulation. For the multi-body pendulum, each body is constrained to rotate about the joints. With rotation matrices chosen as the configuration space, the constraint forces are eliminated from the equations.

2.1 Lagrangian mechanics

In Lagrangian mechanics, the equations of motion are derived with Hamilton's principle. This principle describes the motion of a mechanical system, for which all forces can be derived from scalar potentials. Among all possible paths taken by the system, the correct one must obey Hamilton's principle, which states that the path taken by the system as the one for which the "ac-

tion" of the system is stationary. Here the main ideas of Classical mechanics from [Gol00] and [TLM17] are presented.

Theorem 2.1.1 (Hamilton's Principle)**.** *The trajectory of the mechanical system from time t_0 to time T are extremals of the action integral defined by:*

$$J = \int_{t_0}^{T} \mathcal{L}(t, q, \dot{q}) dt$$

The lagrangian is defined where $\mathcal{L} = K - U$.

The trajectories are minima incurring the "least action".

Given some differentiable curves $q_i(t)$ describing the motion of the system, we consider a family of curves:

$$q_i^\epsilon(t)$$

Where ϵ is some small parameter. The curves $q_i^\epsilon(t)$ are defined such that the satisfy the boundary conditions, meaning $q_i^\epsilon(t_0) = q(t_0)$ and $q_i^\epsilon(T) = q(T)$.

The infinitesimal variation of the curves are:

$$\delta q = \frac{\partial}{\partial \epsilon} q^\epsilon \Big|_{\epsilon=0}$$
$$\delta \dot{q} = \frac{\partial}{\partial \epsilon} \dot{q}^\epsilon \Big|_{\epsilon=0}$$

The action integral for the family of curves are thus:

$$J^\epsilon = \int_{t_0}^{T} \mathcal{L}(t, q^\epsilon, \dot{q}^\epsilon) dt$$

Among the family of varied paths, the one the system takes is the one for which the action integral is stationary. That is the curve for which the infinitesimal variation of the action integral is zero:

$$\int_{t_0}^{T} \mathcal{L}(t, q^\epsilon, \dot{q}^\epsilon) dt$$

From this expression, the Euler-Lagrange equations may be derived.

Hamilton's principle is too restricted to be applied to many real world problems including friction or external forces. These forces may be included by a simple modification to Hamilton's principle - the Lagrange-d'Alemberts principle.

Theorem 2.1.2 (Lagrange-d'Alembert's principle)**.** *The motion of the system from time t_0 to time T is such that*

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

holds for infinitesimal variations consistent with the constraints.

The theorem states that a infinitesimal variation $\delta q(t)$ of the action integral equals the negative of the work done by external forces δW, for a infinitesimal variation of the configuration.

The addition of the term $-\int_{t_0}^{T} \delta W dt$ allows us to include external forces. Gravity interpreted as a applied force rather than as a result of a gravity potential, and may also be included the term on the right hand side. Thus we have a new definition of the Lagrangian:

$$\mathcal{L}(t, q, \dot{q}) = K$$

Before proceeding with deriving the equations of motion from the variational principle, the variation of the generalized coordinates expressed in $SE(3)$ will be derived, and the variation of $SO(3)$ will be obtain as a byproduct.

2.1.1 Variation on SE(3)

In this section, the variation of SE(3) is derived. The derivation is largely taken from [TLM17, p. 316] and [MI19]. Before proceeding with the variation of $SE(3)$, the family of varied curves on $SO(3)$ and $\mathbb{R}^n$ is defined.

Variation on $SO(3)$ A The family of varied paths on $SO(3)$ is defined with the exponential map as:

$$R^{\epsilon} = R(t) \mathrm{e}^{\epsilon \hat{\eta}(t)}, \quad R \in SO(3), \hat{\eta}(t) \in \mathfrak{so}(3)$$

Where $\hat{\eta} : [t_0, T] \to \mathfrak{so}(3)$ is a arbitrary curve that vanishes at the end points $\hat{\eta}(t_0) = \hat{\eta}(T)$.

The infinitesimal variation of the group element follows immediately by application of the definition:

$$\begin{aligned} \delta R^{\epsilon} &= \tfrac{\partial}{\partial \epsilon} R \mathrm{e}^{\epsilon \hat{\eta}(t)} \Big|_{\epsilon=0} \\ &= R\hat{\eta} \end{aligned} \tag{2.1}$$

Variation on $\mathbb{R}^n$ A The family of varied paths on $\mathbb{R}^n$ is defined as:

$$x^\epsilon(t) = x(t) + \epsilon\chi(t), \quad x(t), \chi(t) \in \mathbb{R}^n$$

Where $\chi(t) : [t_0, T] \to \mathbb{R}^n$ is a arbitrary curve that vanishes at the end points $\chi(t_0) = \chi(T)$.

The infinitesimal variation of the group element follows immediately by application of the definition:

$$\begin{aligned} \delta\chi^\epsilon(t) &= \tfrac{\partial}{\partial\epsilon} x(t) + \epsilon\chi(t)\Big|_{\epsilon=0} \\ &= \chi(t) \end{aligned} \tag{2.2}$$

Variation of $SE(3)$ The collection of varied curves on $SE(3)$ is defined by

$$E^\epsilon(t) = E(t)\mathrm{e}^{\epsilon\Gamma}$$

Where Γ denotes a arbitrary curve $\Gamma : [t_0, T] \to \mathfrak{se}(3)$ of the form:

$$\Gamma = \begin{bmatrix} \hat{\eta} & \chi \\ 0 & 0 \end{bmatrix}$$

where $\hat{\eta} : [t_0, T] \to \mathfrak{so}(3)$ and $\chi : [t_0, T] \to \mathbb{R}^3$ vanish at t_0 and T.

The infinitesimal variation of $SE(3)$ is:

$$\delta E = \tfrac{\partial}{\partial\epsilon} E^\epsilon\Big|_{\epsilon=0} = E\Gamma \tag{2.3}$$

Variation of the group tangent Differentiating eq.(2.3)

$$\frac{\mathrm{d}}{\mathrm{d}t}\delta E = \dot{E}\Gamma + E\frac{\mathrm{d}}{\mathrm{d}t}\Gamma = E\left(\Omega\Gamma + \tfrac{\mathrm{d}}{\mathrm{d}t}\Gamma\right)$$

And taking the variation of the kinematic equation $\dot{E} = E\Omega$:

$$\delta\frac{\mathrm{d}}{\mathrm{d}t}E = \delta E\Omega + E\delta\Omega = E\left(\Gamma\Omega + \delta\Omega\right)$$

By equating the two and reworking the terms we obtain:

$$\begin{aligned} \delta\Omega &= \tfrac{\mathrm{d}}{\mathrm{d}t}\Gamma + \Omega\Gamma - \Gamma\Omega \\ &= \tfrac{\mathrm{d}}{\mathrm{d}t}\Gamma + [\Omega, \Gamma] \end{aligned}$$

Collecting and expanding the matrices:

$$\delta\Omega = \begin{bmatrix} \frac{\mathrm{d}}{\mathrm{d}t}\hat{\eta} + [\hat{\omega}, \hat{\eta}] & \frac{\mathrm{d}}{\mathrm{d}t}\delta x + \hat{\omega}\delta x - \hat{\eta}v \\ 0 & 0 \end{bmatrix}$$

The upper left is the variation of $SO(3)$. The upper right block can be further processed. The following result is from [MI19]:

Expanding and collecting terms:

$$\begin{aligned} \delta v &= \tfrac{\mathrm{d}}{\mathrm{d}t}\delta x + \hat{\omega}\delta x - \eta v \\ \delta\left(R^\top \dot{x}\right) &= \tfrac{\mathrm{d}}{\mathrm{d}t}\left(R^\top \delta x\right) + \hat{\omega}R^\top \delta x - \hat{\eta}R^T v \\ R\cdot|\ \delta R^\top \dot{x} + R^\top \delta\dot{x} &= \tfrac{\mathrm{d}}{\mathrm{d}t}R^\top \delta x + R^\top \tfrac{\mathrm{d}}{\mathrm{d}t}\delta x + R^\top \dot{R}R^\top \delta x - R^\top \delta R R^\top v \\ \delta\dot{x} &= \tfrac{\mathrm{d}}{\mathrm{d}t}\delta x + \tfrac{\mathrm{d}}{\mathrm{d}t}\left(RR^\top\right)\delta x - \delta\left(RR^\top\right)v \\ \delta\dot{x} &= \tfrac{\mathrm{d}}{\mathrm{d}t}\delta x \end{aligned}$$

Collecting the results, the final variation of the group tangent is:

$$\begin{bmatrix} \frac{\mathrm{d}}{\mathrm{d}t}\hat{\omega}(t) & \delta v(t) \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \frac{\mathrm{d}}{\mathrm{d}t}\hat{\eta}(t) + \left[\hat{\omega}(t), \hat{\eta}(t)\right] & \frac{\mathrm{d}}{\mathrm{d}t}\delta x(t) \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \frac{\mathrm{d}}{\mathrm{d}t}\hat{\eta}(t) + \hat{\omega}(t)\eta & \frac{\mathrm{d}}{\mathrm{d}t}\delta x(t) \\ 0 & 0 \end{bmatrix} \tag{2.4}$$

Thus the variation in the trajectory of a body undergoing simultaneous rotation and translation is:

$$\delta\hat{\omega}(t) = \tfrac{\mathrm{d}}{\mathrm{d}t}\hat{\eta}(t) + \hat{\omega}(t)\eta \tag{2.5}$$

$$\delta v(t) = \tfrac{\mathrm{d}}{\mathrm{d}t}\delta x(t) \tag{2.6}$$

With the infinitesimal variation for $SE(3)$ and $\mathfrak{se}(3)$ derived, we may proceed with developing the Euler-Lagrange equations.

2.2 Euler-Lagrange equations - Multibody system

In this section the Euler-Lagrange equations of motion for the multibody system will be developed from d'Alembert's principle. The derivation is largely identical to the one outlined in [MI19], with the coordinate-free formulation proposed in [TLM17]. The presentation is more in depth as the formalism applied by Murakami utilizes some special notation.

First we need some definitions.

Definition 2.2.1 (The essential generalized coordinates)**.** The essential generalized coordinates the concatenated configuration variables:

$$[q] = \begin{bmatrix} q^{(1)} \\ \vdots \\ q^{(j)} \\ \vdots \\ q^{(n)} \end{bmatrix} \tag{2.7}$$

The essential generalized velocity and virtual essential displacement are the of the velocities and variations of the generalized coordinates assembled in a vector.

Definition 2.2.2 (The essential generalized velocity and virtual essential displacement)**.**

- The essential generalized velocity vector 3.2 is the concatenation of the velocity of the generalized coordinates.
- The virtual essential displacement vector 3.3 is the concatenation of the variation of the generalized coordinates.

$$\dot{q} = \begin{bmatrix} \dot{q}^{(1)} \\ \vdots \\ \dot{q}^{(j)} \\ \vdots \\ \dot{q}^{(n)} \end{bmatrix} \tag{2.8}$$

$$\delta q = \begin{bmatrix} \delta q^{(1)} \\ \vdots \\ \delta q^{(j)} \\ \vdots \\ \delta q^{(n)} \end{bmatrix} \tag{2.9}$$

The translational and angular velocity in euclidean coordinates are assembled in vector as well.

Definition 2.2.3 (The system generalized velocity and virtual generalized displacement)**.**

- The system generalized velocity vector $\dot{X} \in \mathbb{R}^{6n}$ is the concatenation of the translational velocity in the inertial frame $\dot{x}(t)$ and angular velocity in the local frame $\omega^{(i/i-1)}$ of the bodies in the system.
- The virtual generalized displacement vector $\delta X \in \mathbb{R}^{6n}$ is the concatenation of the variation in the trajectory in the inertial frame $\dot{x}(t)$ and variation of rotation in the local frame $\omega^{(i/i-1)}$ of the bodies in the system.

$$\dot{X} = \begin{bmatrix} \underline{v}^{(1)} \\ \underline{\omega}^{(1)} \\ \vdots \\ \underline{\omega}^{(j)} \\ \underline{v}^{(j)} \\ \vdots \\ \underline{\omega}^{(n)} \\ \underline{v}^{(n)} \end{bmatrix} \tag{2.10}$$

$$\delta X = \begin{bmatrix} \delta\underline{x}^{(1)} \\ \underline{\eta}^{(1)} \\ \vdots \\ \delta\underline{x}^{(j)} \\ \underline{\eta}^{(j)} \\ \vdots \\ \delta\underline{x}^{(n)} \\ \underline{\eta}^{(n)} \end{bmatrix} \tag{2.11}$$

A central part to Murakamis solution to the Euler-Lagrange equations, is that the system generalized velocity and virtual generalized displacement are linearly related to the essential generalized velocity and essential virtual displacement respectively. So it is possible to write:

$$\delta X = B\delta q \tag{2.12}$$

$$\dot{X} = B\dot{q} \tag{2.13}$$

Variation of system generalized velocity The variation of the system generalized velocity **??**

$$(\delta\hat{\omega}(t), \delta v(t)) = (\frac{\mathrm{d}}{\mathrm{d}t}\hat{\eta}(t) + \hat{\omega}(t)\eta, \frac{\mathrm{d}}{\mathrm{d}t}\delta x(t))$$

for the system can be rewritten as a block matrix-vector product

$$\delta\dot{X} = \frac{\mathrm{d}}{\mathrm{d}t}\delta X + D\delta X, \tag{2.14}$$

where

$$D = \begin{bmatrix} 0_{3\times 3} & & & \\ & \widehat{\omega^{(1)}} & & \\ & & \ddots & \\ & & & 0_{3\times 3} \\ & & & & \widehat{\omega^{(n)}} \end{bmatrix} \tag{2.15}$$

The external forces are assembled in the force vector.

Definition 2.2.4 (Force vector)**.** The force vector is defined as external forces and moments including gravity concatenated in a vector:

$$\underline{F} = \begin{bmatrix} f^{(1)} \\ M^{(1)} \\ \vdots \\ f^{(n)} \\ M^{(n)} \end{bmatrix} \tag{2.16}$$

where forces are applied in the inertial frame, while moments are applied in the local frame.

Virtual work The work performed by the external forces and moments 2.16 through a infinitesimal displacement consistent with the constraints can be expressed together with the definition of the virtual generalized displacement as follows. In engineering terms, this is the "virtual work".

$$\delta W(t) = \{\delta X(t)\}^\top F \tag{2.17}$$

Kinetic energy With the definition of the system generalized velocity, the kinetic energy of the system is easily expressed by the quadratic form:

$$K(t) = \frac{1}{2}\dot{X}^\top M\dot{X} \tag{2.18}$$

Where M is the symmetric positive definite block-matrix:

$$M = \begin{bmatrix} m_1 I_3 & & & \\ & J_1 & & \\ & & \ddots & \\ & & m_n I_3 & \\ & & & J_n \end{bmatrix}, \tag{2.19}$$

where J_n are the moment of inertia tensors around the centers of mass of the bodies.

2.2.1 Deriving the equations of motion

Now we have the necessary definitions to proceed with deriving the equations of motion. Stating d'Alembert-Lagrange principle and inserting the variation in kinetic energy and the virtual work.

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

Remembering the symmetry of the mass-matrix M, the variation of the kinetic energy is:

$$\delta K(t) = \delta \dot{X}^\top M \dot{X}$$
$$\int_{t_0}^{T} \delta \dot{X}^\top M \dot{X} + \{\delta X(t)\}^\top F dt = 0$$

For the variation of the system generalized velocity 2.14 is inserted.

$$\int_{t_0}^{T} \{\frac{\mathrm{d}}{\mathrm{d}t}\delta X + D\delta X\}^\top M \dot{X} + \{\delta X(t)\}^\top F dt = 0$$

Then performing the integration by parts and enforcing the border-conditions on the variation yields:

$$\int_{t_0}^{T} \{\delta X\}^\top \left(-\tfrac{\mathrm{d}}{\mathrm{d}t} \left(M\dot{X} \right) - DM\dot{X} + F \right) dt = 0$$

The δX terms are now rewritten. We insert 2.12 and2.13 and reorder:

$$-\int_{t_0}^{T} \{\delta q\}^\top B^\top \left(\tfrac{\mathrm{d}}{\mathrm{d}t} \left(MB\dot{q} \right) + DMB\dot{q} - F \right) dt = 0$$
$$-\int_{t_0}^{T} \{\delta q\}^\top B^\top \left(MB\ddot{q} + (M\dot{B} + DMB)\dot{q} - F \right) = 0$$

Yields the Euler-Lagrange equations:

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

Which is restated:

$$\begin{cases} M^*(t)\ddot{q} + N^*(t)\dot{q} = F^* \\ M^*(t) = B^\top MB \\ N^*(t) = B(t)^\top (M\dot{B}(t) + D(t)MB(t)) \\ F^*(t) = B(t)^\top F(t) \end{cases} \tag{2.20}$$

2.2.2 Euler-Lagrange equations for coordinate-free N-body pendulum

Whereas Murakami et. al. determines the equations in terms of Euler angles, we will follow a different approach inspired by Leok et. al. Instead of Euler angles we will derive equations in terms of the direct representation of $SO(3)$. The bodies in the n-body pendulum are restricted to rotate about the joints, thus by taking the rotation matrices as the orientation relative to the parenting frame, the essential generalized coordinates for the n-body pendulum are.

$$q(t) = \begin{bmatrix} R^{(1)}(t) \\ \vdots \\ R^{(k/k-1)}(t) \\ \vdots \\ R^{(n/n-1)}(t) \end{bmatrix}$$

The essential generalized velocities 3.2 and the essential virtual generalized displacements 3.3 for the N-body pendulum are:

$$\dot{q} = \begin{bmatrix} \underline{\omega}^{(1)}(t) \\ \vdots \\ \underline{\omega}^{(k/k\,-\,1)}(t) \\ \vdots \\ \underline{\omega}^{(n/n\,-\,1)}(t) \end{bmatrix}, \quad \in \mathbb{R}^{(3n)} \quad (2.21) \qquad \delta q = \begin{bmatrix} \underline{\eta}^{(1)}(t) \\ \vdots \\ \underline{\eta}^{(k/k\,-\,1)}(t) \\ \vdots \\ \underline{\eta}^{(n/n\,-\,1)}(t) \end{bmatrix}, \quad \in \mathbb{R}^{(3n)} \tag{2.22}$$

where $\hat{\omega}^{(i/i\,-\,1)}(t) \in \mathfrak{so}(3), \hat{\eta}^{(i/i\,-\,1)} \in \mathfrak{so}(3) \quad i = 1, \ldots, n.$

A insufficiency of the notation of the moving frame formalism is the choice of $\dot{q}$ denoting the essential generalized velocities. When the configuration is represented by multiple rotation matrices, the $\dot{q}$ notation would imply the velocity is represented by the derivative of the rotation-matrix, but this is not the case. As we will see in the next section, the most convenient representation of the essential generalized velocities for Lie group representations is at the group identity (Lie Algebra).

To emphasize that we for the N-body pendulum are expressing the essential generalized virtual velocity and the essential virtual generalized displacement in the Lie algebra of $SO(3)$, the notation for the generalized velocities will be changed to:

$$\dot{q} \Rightarrow \boldsymbol{\omega}$$
$$\delta q \Rightarrow \boldsymbol{\eta}$$

This makes sense for systems with all generalized coordinates expressed in $SO(3)$ rotation-matrices. Thus Ω and H without superscripts or hyphenation are the essential generalized velocities and virtual essential displacements of the system with this new notation, the system is rewritten as:

$$\begin{cases} M^*(t)\dot{\boldsymbol{\omega}} + N^*(t)\boldsymbol{\omega} = F^* \\ M^*(t) = B^\top(t) M B(t) \\ N^*(t) = B(t)^\top (M\dot{B}(t) + D(t) M B(t)) \\ F^*(t) = B(t)^\top F(t) \end{cases} \tag{2.23}$$

Where, $B(t)$ and $\dot{B}(t)$ are dependent on the generalized coordinates. This system evolves on n copies of the group of rotation matrices $(SO(3))^n$.

2.3 B - Matrix

In this section, the B-matrix in the following relation will be developed:

$$\delta X(t) = B(t)\boldsymbol{\eta}$$
$$\dot{X}(t) = B(t)\boldsymbol{\omega}$$

The entries of the B-matrix will be determined by reworking the the right-hand side of kinematic equations for the n-body pendulum that were derived in 1.4. Supressing the time-dependency, the system generalized coordinates are:

$$\begin{bmatrix} \underline{x}_1(t) \\ R^{(1)} \\ \vdots \\ \underline{x}_n(t) \\ R^{(n/n-1)} \end{bmatrix} = \begin{bmatrix} R^{(1)}(t)\underline{s}_{1,1} \\ R^{(1)} \\ \vdots \\ \underline{x}_{n-1}(t) + R^{(n-1)}(t)\underline{s}_{n-1,2} + R^{(n)}(t)\underline{s}_{n,1} \\ R^{(n/n-1)} \end{bmatrix}$$

The basis of multibody dynamics formulation in the MFM is the fact that the relation between variation and velocity in euclidean space and the configuration manifold exists, and are equivalent. This is intuitively clear considering that both variation and time-differentiation involve taking the derivative of the equations expressing the configuration. $\frac{\partial}{\partial\epsilon}x(t+\epsilon)\Big|_{\epsilon=0}$ and $\frac{\mathrm{d}}{\mathrm{d}t}x(t)$ ammounts to taking the derivative of the time-dependent terms. This common relation is what will become the B-matrix. In the following section, the B-matrix will be developed for a both cases. First the translation terms $\underline{x}(t)$ will be handled, then the angular terms ω, before the expressions are assembled to form the B-matrix.

Supressing the notation for the time-dependency. From 1.2 we had obtained eq.(1.10).

$$\frac{\mathrm{d}}{\mathrm{d}t}R^{(k)} = R^{(k)}\left(\sum_{j=1}^{k}\widetilde{R^{(k/j)^\top}\underline{\omega}^{(j/j-1)}}\right)$$

Likewise, it is easily seen that the variation of the same expression is:

$$\delta R^{(k)} = R^{(k)}\left(\sum_{j=1}^{k}\widetilde{R^{(k/j)^\top}\underline{\eta}^{(j/j-1)}}\right) \tag{2.24}$$

2.3.1 Translative velocity terms

From the section 1.4 we had position and orientation of a joint in the multi-body system expressed by eq.(1.12): The first term is simple:

$$\begin{aligned} \delta x(t) &= R\hat{\eta}s_i \\ &= -R\hat{s}_i\eta \\ \tfrac{\mathrm{d}}{\mathrm{d}t}x(t) &= R\hat{\omega}s_i \\ &= -R\hat{s}_i\omega \end{aligned}$$

The position for the k'th body is:

$$\frac{\mathrm{d}}{\mathrm{d}t}x_k(t) = \frac{\mathrm{d}}{\mathrm{d}t}x_{k-1} + \frac{\mathrm{d}}{\mathrm{d}t}\left(R^{(k-1)}\underline{s}_{k-1,2} + R^{(k)}\underline{s}_{k,1}\right) \tag{2.25}$$

Looking closer to the term $\frac{\mathrm{d}}{\mathrm{d}t}R^{(k)}\underline{s}_{k,1}$, we apply eq.(1.10) and rework the expression.

$$\begin{aligned} \tfrac{\mathrm{d}}{\mathrm{d}t}R^{(k)}\underline{s}_{k,i} &= R^{(k)}\left(\overbrace{\sum_{j=1}^{n} {R^{(k/j)}}^{\top}\underline{\omega}^{(j/j-1)}}\right)\underline{s}^{(k)} \\ &= -R^{(k)}\hat{s}^{(k)}\left(\sum_{j=1}^{k} {R^{(k/j)}}^{\top}\underline{\omega}^{(j/j-1)}\right) \end{aligned}$$

With this result, eq.(2.25) can be rewritten:

$$\begin{aligned} \tfrac{\mathrm{d}}{\mathrm{d}t}x_k(t) = \tfrac{\mathrm{d}}{\mathrm{d}t}x_{k-1}(t) &- R^{(k-1)}\hat{s}_{(k-1,2)}\left(\sum_{j=1}^{k-1} {R^{(k-1/j)}}^{\top}\underline{\omega}^{(j/j-1)}\right) \\ &- R^{(k)}\hat{s}_{(k,1)}\left(\sum_{j=1}^{k} {R^{(k/j)}}^{\top}\underline{\omega}^{(j/j-1)}\right) \end{aligned} \tag{2.26}$$

Which is the expression we were looking for.

To clarify the construction of the B-matrix a new notation will be established. Given the k first relative rotation matrices $R^{(k/1)}, \ldots, R^{(k/k-1)}$ of the n-body pendulum, the rotation matrices are concatenated in a block column

matrix:

$$\mathcal{R}_k = \begin{bmatrix} R^{(k/1)} \\ \vdots \\ R^{(k/j-1)} \\ \vdots \\ I \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \in \mathbb{R}^{3\times 3n} \tag{2.27}$$

Using the definitions, eq.(2.25) can be reformulated. The first body is the simplest:

$$\begin{aligned} \tfrac{\mathrm{d}}{\mathrm{d}t} x_1(t) &= -R^{(1)}\hat{s}_i\omega^{(1)} \\ &= B_1^{transl}\omega, \quad B_1^{transl} = -R^{(1)}\hat{s}_i \end{aligned}$$

Where $B_{1,t}$ is the linear relation between translation and generalized coordinates. The first index denotes the body, and the second denotes that it relates generalized coordinates to translation. With body-1 defined, the second body is.

$$\begin{aligned} \tfrac{\mathrm{d}}{\mathrm{d}t}\underline{x}^{(2)} =& (B_1^{transl} + (-R^{(1)}\hat{s}_{1,i}^{(1)}\mathcal{R}_{(1)}^{\top}) \\ &+ (-R^{(2)}\hat{s}_{2,i}^{(2)}\mathcal{R}_2^{\top}))\boldsymbol{\omega} \\ =& B_2^{transl}\boldsymbol{\omega} \end{aligned}$$

The k'th body.

$$\begin{aligned} \tfrac{\mathrm{d}}{\mathrm{d}t}\underline{x}_k(t) =& (B_{k-1}^{transl} + \left(-R^{(k-1)}\hat{s}_{k-1,i}^{(k-1)}\mathcal{R}_{k-1}^{\top}\right) \\ &+ \left(-R^{(k)}\hat{s}_{k,i}^{(k)}\mathcal{R}_k^{\top}\right))\boldsymbol{\omega} \qquad (2.28) \\ =& B_k^{transl}\boldsymbol{\omega} \qquad (2.29) \end{aligned}$$

Which neatly encapsulates the relation between time derivative in configuration to time-derivative in euclidean space of the k'th body for a pendulum constraint.

Similarily for a variation in $\underline{x}_k(t)$ we apply eq.(2.24) and rework the terms and obtain the expression:

$$\delta\underline{x}_k(t) = B_k^{transl}\boldsymbol{\eta}$$

2.3.2 Rotational terms

Derivation is a simpler. Formulating the relation between the absolute angular velocity in terms of relative angular velocity skew matrices is straightforward with the already established notation.

$$\hat{\omega}^{(k)} = \sum_{j=1}^{k} \overbrace{R^{(k/j)^\top} \underline{\omega}^{(j/j-1)}}$$

The vector representation is then simply:

$$\underline{\omega}^{(k)} = \sum_{j=1}^{k} R^{(k/j)^\top} \underline{\omega}^{(j/j-1)}$$

Unlike the translation terms the angular terms are expressed in the local frames.

Thus we have the common relation:

$$\omega^{(k)} = \mathcal{R}_k^\top \boldsymbol{\omega} = B_k^{rot} \boldsymbol{\omega}$$
$$\eta^{(k)} = \mathcal{R}_k^\top \boldsymbol{\eta} = B_k^{rot} \boldsymbol{\eta}$$

Where $B_k^{rot} = \mathcal{R}_k^\top$ is the relation between velocity in generalized coordinates and local angular velocity. The first index denotes the body and the second denotes that it relates absolute generalized velocity and a variation of rotation in the inertial frame to essential generalized velocities and virtual essential displacements.

2.3.3 Final assembly

With the translational- and angular-velocity terms for the bodies expressed as a expression which is linear in the generalized coordinates, the B-matrix is ready to be assembled for the n-body pendulum. With the translational- and angular- velocity for the first body:

$$\tfrac{\mathrm{d}}{\mathrm{d}t} x^{(1)}(t) = -\mathcal{R}_{(1)} \hat{s}_{1,1} \boldsymbol{\omega}$$
$$\omega^{(1)} = \mathcal{R}_{(1)}^\top \boldsymbol{\omega}$$

and for the k'th body

$$\tfrac{\mathrm{d}}{\mathrm{d}t} x^{(k)}(t) = \tfrac{\mathrm{d}}{\mathrm{d}t} x_{j-1} - R^{(k-1)} \hat{s}_{k-1,2} \mathcal{R}_{k-1}^\top \boldsymbol{\omega} - R^{(k)} \hat{s}_{k,1} \mathcal{R}_k^\top \boldsymbol{\omega}$$
$$\omega^{(k)}(t) = \mathcal{R}_k^\top \boldsymbol{\omega}$$

By assembling these terms into a vector and a block matrix, we can write:

$$\begin{bmatrix} \frac{\mathrm{d}}{\mathrm{d}t}x^{(1)}(t) \\ \omega^{(1)}(t) \\ \vdots \\ \frac{\mathrm{d}}{\mathrm{d}t}x^{(k)}(t) \\ \omega^{(k)}(t) \\ \vdots \\ \frac{\mathrm{d}}{\mathrm{d}t}x^{(n)}(t) \\ \omega^{(n)}(t) \end{bmatrix} = \begin{bmatrix} -\mathcal{R}_{(1)}\hat{s}_{1,1} \\ \mathcal{R}_{(1)}^{\top} \\ \vdots \\ B_{k-1}^{transl} - R^{(k-1)}\hat{s}_{k-1,2}\mathcal{R}_{k-1}^{\top} - R^{(k)}\hat{s}_{k,1}\mathcal{R}_{k}^{\top} \\ \mathcal{R}_{(k)}^{\top} \\ \vdots \\ B_{n-1}^{transl} - R^{(n-1)}\hat{s}_{n-1,2}\mathcal{R}_{n-1}^{\top} - R^{(n)}\hat{s}_{n,1}\mathcal{R}_{n}^{\top} \\ \mathcal{R}_{(n)}^{\top} \end{bmatrix} \omega \tag{2.30}$$

Which is the relation we were looking for.

$$\dot{X} = B\boldsymbol{\omega}, where B(R^{(1)}, R^{(2/1)}, \ldots, R^{(n/n-1)})$$

where

$$B = \begin{bmatrix} B_1^{transl} \\ B_1^{rot} \\ \vdots \\ B_k^{transl} \\ B_k^{rot} \\ \vdots \\ B_n^{transl} \\ B_n^{rot} \end{bmatrix}$$

The matrix can be generated iteratively, starting from the first body. Doing the same process for the variation δX results in:

$$\delta X(t) = B\boldsymbol{\eta}, where B(R^{(1)}, R^{(2/1)}, \ldots, R^{(n/n-1)})$$

2.4 $\dot{B}$ matrix

In the Euler-Lagrange equations eq.(2.23), the second term on the left hand side involves the time derivative of the B-matrix. This $\dot{B}$-matrix will be closer examined in this section.

For applications formulated where the B-matrix is expressed symbolically, the most straightforward method to determine $\dot{B}$ is to take the derivative of the entries in the B-matrix. But in the case of the coordinate-free representation, we do not have such a explicit expression. In this section we instead seek to determine a iterative algorithm for the $\dot{B}$ in a similar vein to the previous section.

Sadly, attempting to differentiate the expressions for the block entries composing the B-matrix from the previous section directly yields poor results. For large systems, the entries of the B-matrix consists of a products of a large number of rotation matrices. Thus the differentiating operation would produce a n-term sum in each block-entry of the $\dot{B}$-matrix for a n-body system. When extending the number of bodies in the multi-body system this would cause a cubic growth in computation cost. Therefore to obtain a better expression for we recall that the B-matrix was derived by rewriting expressions consisting of the adjoint operator applied to the essential generalized velocities.

$$Ad_{R^{n/j}}\left(\hat{\xi}\right) = {R^{(n/j)}}^{\top}\hat{\xi}R^{(n/j)} \quad \hat{\xi} \in \mathfrak{so}(3)$$

To find the $\dot{B}$ matrix, we seek to find a representation of the time derivative of the adjoint operator, $\frac{\mathrm{d}}{\mathrm{d}t}Ad_{R^nj}(\hat{\xi})$, and rewriting the resulting expressions. The derivative of the adjoint terms can be formulated as follows:

Theorem 2.4.1 (The derivative of the adjoint term)**.** *In the context of the thesis, the derivative of the adjoint operator of $SO(3)$ is:*

$$\frac{\mathrm{d}}{\mathrm{d}\tau}Ad_{R^{(n/j)}(t+\tau)}\hat{\xi}\Big|_{\tau=0} = -\widehat{\hat{\omega}^{(n/j)}R^{(n/j)}\hat{\xi}} \tag{2.31}$$

Proof. Consider the curve $R(t+\tau) \in SO(3)$:

$$\begin{aligned} R(t+\tau) &= R(t) + \dot{R}(t)\tau + \cdots \\ &= R(I + \hat{\omega}\tau + \cdots) \\ R^{\top}(t+\tau) &= \left(I - \hat{\omega}\tau + \mathcal{O}(t^2)\right)R^{\top} \end{aligned}$$

The adjoint operator may be rewritten as follows:

$$\begin{aligned} Ad_{R^{(n/j)}(t+\tau)}(\hat{\xi}) &= \left(e - \hat{\omega}^{(n/j)}\tau + \mathcal{O}(\tau^2)\right) {R^{(n/j)}}^{\top} \hat{\xi} R^{(n/j)} \left(e + \hat{\omega}^{(n/j)}\tau + \mathcal{O}(\tau^2)\right) \\ &= {R^{(n/j)}}^{\top} \hat{\xi} R^{(n/j)} - \hat{\omega}^{(n/j)}\tau {R^{(n/j)}}^{\top} \hat{\xi} R^{(n/j)} \\ &\qquad + {R^{(n/j)}}^{\top} \hat{\xi} R^{(n/j)} \hat{\omega}^{(n/j)}\tau + \mathcal{O}(\tau^2) \\ &= Ad_{R^{(n/j)}}\left(\hat{\xi}\right) - \left[\hat{\omega}^{(n/j)}, Ad_{R^{(n/j)}}(\hat{\xi})\right]\tau + \mathcal{O}(\tau^2) \end{aligned}$$

Then taking the derivative at $\tau = 0$:

$$\tfrac{\mathrm{d}}{\mathrm{d}\tau} Ad_{R^{(n/j)}(t+\tau)}\hat{\xi}\Big|_{\tau=0} = -\left[\hat{\omega}^{(n/j)}, Ad_{R^{(n/j)}}\left(\hat{\xi}\right)\right]$$

The adjoint term within the commutator may be rewritten using eq.(1.5).

$$-\left[\hat{\omega}^{(n/j)}, Ad_{R^{(n/j)}}\left(\hat{\xi}\right)\right] = -\left[\hat{\omega}^{(n/j)}, \widehat{{R^{(n/j)}}^{\top}\underline{\xi}}\right]$$

and finally applying $[\hat{\xi}_1, \hat{\xi}_2] = \widehat{\hat{\xi}_1 \xi_2}$ we arrive at the final expression of the term.

$$\tfrac{\mathrm{d}}{\mathrm{d}\tau} Ad_{R^{(n/j)}(t+\tau)}\left(\hat{\xi}\right)\Big|_{\tau=0} = -\widehat{\hat{\omega}^{(n/j)} R^{(n/j)} \hat{\xi}}$$

□

Thus, the differentiation of the adjoint operation is reduced to the evaluation of a single skewed product rather than a sum of term, which is a significant reduction in computation effort.

2.4.1 Constructing $\dot{B}$: Translation terms

For the rows translation terms of we had:

$$\frac{\mathrm{d}}{\mathrm{d}t}x^{(k)}(t) = \frac{\mathrm{d}}{\mathrm{d}t}x^{(k-1)}(t) - R^{(k-1)}\left(\sum_{j=1}^{k-1} Ad_{R^{(k\,-\,1/j)}}(\hat{\omega}^{(i/i\,-\,1)})\underline{s}_{k-1,1}\right) - R^{(k)}\left(\sum_{j=1}^{k} Ad_{R^{(k/j)}}(\hat{\omega}^{(i/i\,-\,1)})\underline{s}_{k,1}\right)$$

Taking the derivative of one term and using eq.(2.31) we obtain:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}R^{(n)}Ad_{R^{(n/j)}}\left(\hat{\omega}^{(j/j\,-\,1)}\right) &= \dot{R}^{(n)}Ad_{R^{(n/j)}}\left(\hat{\omega}^{(j/j\,-\,1)}\right) + R^{(n)}\frac{\mathrm{d}}{\mathrm{d}t}Ad_{R^{(n/j)}}\left(\hat{\omega}^{(j/j\,-\,1)}\right) \\ &= R^{(n)}\left(\overbrace{\hat{\omega}^{(n)}R^{(n/j)^{\top}}\omega^{(j/j\,-\,1)}} - \overbrace{\hat{\omega}^{(n/j)}R^{(n/j)^{\top}}\omega^{(j/j\,-\,1)}}\right) \qquad (2.32)\end{aligned}$$

Then we can reformulate the terms in the velocity expression:

$$\frac{\mathrm{d}}{\mathrm{d}t}R^{(k)}Ad_{R^{(n/j)}}(\hat{\omega}^{(j/j\,-\,1)})\underline{s}_{j,i} = - R^{(k)}\left(\hat{\omega}^{(k)}\hat{s}_{k,i} - \hat{s}_{k,i}\hat{\omega}^{(k/j)}\right)R^{(k/j)^{\top}}\underline{\omega}^{(j/j\,-\,1)} \qquad (2.33)$$

Now we may finally turn to the expression for the translation row. We had that the translational velocity of body k in terms of the $[B]$-matrix and essential generalized velocities were:

$$\begin{aligned}\dot{x}^{(k)}(t) &= B_k^{transl}\omega \\ &= B_{k-1}^{transl}\omega - R^{(k-1)}\left(\sum_{j=1}^{k-1} Ad_{R^{(k\,-\,1/j)}}(\hat{\omega}^{(i/i\,-\,1)})\underline{s}_{k-1,2}\right) \\ &\quad - R^{(k)}\left(\sum_{j=1}^{k} Ad_{R^{(k/j)}}(\hat{\omega}^{(j/j\,-\,1)})\underline{s}_{k,1}\right)\end{aligned}$$

Taking the derivative of the adjoints, and inserting eq.(2.33).

$$\dot{B}_k^{transl}\omega = (\tfrac{\mathrm{d}}{\mathrm{d}t}B_k^{transl})\omega$$
$$-R^{(k-1)}\sum_{j=1}^{k-1}\left(\left(\hat{\omega}^{(k-1)}\hat{s}_{k-1,2} - \hat{s}_{k-1,2}\hat{\omega}^{(k-1/j)}\right)R^{(k-1/j)^\top}\underline{\omega}^{(j/j-1)}\right)$$
$$-R^{(k)}\sum_{j=1}^{k}\left(\left(\hat{\omega}^{(k)}\hat{s}_{k,1} - \hat{s}_{k,1}\hat{\omega}^{(k/j)}\right)R^{(k/j)^\top}\underline{\omega}^{(j/j-1)}\right)$$

This expression for the translational rows of the $\dot{B}$ matrix has no compact expression like for the rows of the B-matrix. So we will have to settle for a expression of the 3×3 block entries of the $\dot{B}$-matrix. Given the$[\dot{B}]^l_{k-1,j}$ term, we may write the following:

$$\begin{aligned}\dot{B}_{k,j}^{transl} =& \dot{B}_{k-1,j}^{transl} - R^{(k-1)}\left(\hat{\omega}^{(k-1)}\hat{s}_{k-1,2} - \hat{s}_{k-1,2}\hat{\omega}^{(k-1/j)}\right)R^{(k-1/j)^\top}\\ &- R^{(k)}\left(\hat{\omega}^{(k)}\hat{s}_{k,1} - \hat{s}_{k,1}\hat{\omega}^{(k/j)}\right)R^{(k/j)^\top}\end{aligned} \tag{2.34}$$

This are the block entries for the joints of the solid n-body pendulum.

2.4.2 Angular terms

The rows relating absolute to relative rotation on the B-matrix simply the adjoint $Ad_{R^{(k)}}(\hat{\xi})$.

$$\tfrac{\mathrm{d}}{\mathrm{d}\tau}R^{(k)}(t+\tau)\Big|_{\tau=0} = \sum_{j=1}^{k} Ad_{R^{(k/j)}(t)}\hat{\xi}$$

Applying eq.(2.31) yields:

$$\tfrac{\mathrm{d}}{\mathrm{d}\tau}\sum_{j=1}^{k} Ad_{R^{(k/j)}(\tau)}(\hat{\omega}^{j/j-1})$$
$$=\sum_{j=1}^{k} \overbrace{-\hat{\omega}^{(k/j)}R^{(k/j)}\underline{\omega}^{(j/j-1)}}$$

This is again the skew symmetric form for which the vector representation is:

$$\sum_{j=1}^{k} -\hat{\omega}^{(k/j)} R^{(k/j)} \underline{\omega}^{(j/j-1)}$$

This has a compact matrix representation:

$$\dot{B}_k^{rot} = [-\hat{\omega}^{(k/1)} R^{(k/1)\top}, \cdots, -\hat{\omega}^{(j/1)} R^{(j/1)\top}, 0, \cdots, 0]$$

2.4.3 Final Assembly

Unlike the B-matrix, the $\dot{B}$ matrix does not have a expression for the entire rows associated with the k'th body. So we settle for a expression for the j'th block-collumn for the k'th body.

$$\begin{bmatrix} \dot{B}_{k,j}^{transl} \\ \\ \\ \dot{B}_{k,j}^{rot} \end{bmatrix} = \begin{bmatrix} \dot{B}_{(k-1,j),l} + \cdots \\ -R^{(k-1)} \left(\hat{\omega}^{(k-1)} \hat{s}_{k-1,2} - \hat{s}_{k-1,2} \hat{\omega}^{(k-1/j)} \right) R^{(k-1/j)\top} + \cdots \\ \cdots - R^{(k)} \left(\hat{\omega}^{(k)} \hat{s}_{k,1} - \hat{s}_{k,1} \hat{\omega}^{(k/j)} \right) R^{(k/j)\top} \\ -\hat{\omega}^{(j/1)} R^{(j/1)\top} \end{bmatrix} \tag{2.35}$$

The full $\dot{B}$ is composed by the block-expressions. The matrix is a function of the essential generalized coordinates and velocities.

2.5 Final form of the equations

With the B- and $\dot{B}$-matrices obtained, all the components of equations 2.23 are defined and a solution to the equations may be computed.

Contrary to the initial worry that the expression for the $\dot{B}$-matrix would entail large sums, it can be constructed by much smaller computations. This has great practical implications since extending the multibody system with more links increases the cost quadratically and not cubically as initially feared.

In most treatments on the subject of multibody dynamics in classical mechanics, the expressions for the differential equations are developed explicitly. Then the explicit equations are coded in a appropriate code language and passed to a ODE-solver. Aside from the problems associated with the singularities in the resulting equations, this approach encounter some difficulties when the system grows large. The resulting equations are long expressions of sines and cosines, which grow very big for systems longer than 4 bodies. This effectively puts a cap on the length of the systems that can be simulated. In the thesis [Ryk18] it was shown that the cap lies at about 6 links for a implementation with the MatLab symbolic manipulator. At this point, the expressions were so large that they took 15GB of RAM, filling the working memory of the computer and crashing the program.

Due to the challenges associated with exploding expression, a algorithmic approach to the generation of the B-matrices is proposed based on the expressions 2.30 and 2.35. The arguments of the algorithm is the generalized coordinates $R^{(j/j-1)},\ j = 1, \ldots, n$ and the generalized velocities $\omega^{(j/j-1)},\ j = 1, \ldots, n$. The limitations of this approach is the speed at which the processor is able to generate the matrices. Memory-limitations will only be a problem for extremely large systems. With such a scheme being defined and the M and D constructed according to the definition, the computation of the system may be integrated with a integration scheme.

Chapter 3

Numerical Simulations

From the previous section we had the equations of motion for the N-body pendulum expressed by the system of equations 2.23 in terms of the direct representation of $SO(3)$:

$$M^*(R^{(j/j-1)}(t))\dot{\boldsymbol{\omega}} + N^*(R^{(j/j-1)}(t), \omega^{(j/j-1)}(t))\boldsymbol{\omega} = F^*, \quad j = 1, \dots n$$

The goal of this section is to compute numerical solutions to the Euler-Lagrange equations. To be able to compute a solution, the equations must be rewritten to a form to which we can apply standard ODE-solvers. Scientific computation packages are available that can be applied to directly to systems such as the above (for example in matlab). However, we want methods that preserve some particular system quantities. Therefore we will restate the equations on standard form as a first-order system of differential equations. When the equations are on standard form, we may apply solvers which conserve the quantities in question.

$$\dot{y} = f(t, y)$$

Computing the dynamics of system with direct representation of the $(SO(3))^n$ group poses som practical challenges when attempting numerical experiments. Rounding errors introduced during numerical simulation will at some point break the orthogonality of the matrices, and re-orthogonalizing the matrices is computationally costly. The matrices themselves are $\mathbb{R}^{3\times3}$ and allocating 9 numbers for every coordinate is memory consuming. Furthermore, storing the orientation in as a full 9-element matrix is unnecessary as $SO(3)$ has dimension 3 and can be represented by a minimum of 3 parameters. We wish to avoid the euler-coordinates due to the singularities in the resulting equations of motion, and instead turn to

the unit-quaternions. By utilizing the homomorphism between $SO(3)$ and $\mathbb{S}^3$, we the equations of motion will be rewritten in terms of the direct representation of the unit quaternions. The unit quaternions are a more compact representation of rotations, and has the additional advantage of requiring only a simple normalization to correct for errors. The application of the unit quaternions to describe the kinematics of rigid bodies are not new (see [CR99]), the novelty lies in the application of quaternions to multi-body systems under the MFM formalism.

To solve for $\dot{\boldsymbol{\omega}}$ the M^* matrix must be inverted, and a appropriate numerical scheme will be selected to perform this inversion. To compute a solution to the equations which is as accurate as possible, we will select a ODE-solver which preserves the properties of the system.

A number of simulations for systems of various size are computed in order to assess the accuracy of the computed solutions. The coordinate-free system is compared to a one- and two- body classical pendulum implemented with parametrizing angles. Then the conservation of system quantities is computed.

Toward the end of the chapter some models described by the coordinate-model are simulated. A heavy symmetric top with one point, and some N-body pendulums with external springs and dampers are computed.

3.1 Rewriting the equations

To solve for ω from 2.20, M^* is inverted, and we obtain the expression.

$$\dot{\boldsymbol{\omega}} = M^{*-1}(-N^*\boldsymbol{\omega} + F^*) \tag{3.1}$$

These equations are in terms of the direct representation of $SO(3)$. To rewrite the equations we insert the homomorphisms $R(q) : \mathbb{S}^3 \to SO(3)$ 1.19, and $\xi : T_I\mathbb{S}^3 \to \mathfrak{so}(3)$ 1.5.4 into the equations of motion and obtain a new set of equations in terms of quaternions. Identifying the angular velocity vector with the the complex part of the quaternion-tangent we can write:

$$\xi^{(i/i-1)} = \begin{bmatrix} 0 \\ \frac{1}{2}\omega^{(i/i-1)} \end{bmatrix} \in \mathbb{H}$$

Then the essential generalized coordinates and the essential generalized velocities in terms of quaternions are

$$q = \begin{bmatrix} u^{(1)} \\ \vdots \\ u^{(n/n\,-\,1)} \end{bmatrix} \tag{3.2}$$

$$\boldsymbol{\xi} = \begin{bmatrix} \xi^{(1)} \\ \vdots \\ \xi^{(n/n\,-\,1)} \end{bmatrix} \tag{3.3}$$

The angular velocities in $T_I\mathbb{S}^3$ are pure complex quaternions. Thus by keeping only the complex part, we may write

$$\boldsymbol{\xi}_v = \begin{bmatrix} \xi_v^{(1)} \\ \vdots \\ \xi_v^{(n/n\,-\,1)} \end{bmatrix}$$

With these definitions eq.(3.1) may be rewritten in terms of unit-quaternions.

$$\dot{\boldsymbol{\xi}}_v = 2 \cdot M^*(R^{(i/i\,-\,1)})^{-1}(-N^*(R^{(i/i\,-\,1)}, \hat{\omega}^{(i/i\,-\,1)})\boldsymbol{\omega} + F^*) \tag{3.4}$$

where the homomorphisms

- $R^{(j/j\,-\,1)} = R(u^{(j/j\,-\,1)}) = I_3 + 2u_0^{(j/j\,-\,1)}\hat{u}_v^{(j/j\,-\,1)} + 2\left(\hat{u}_v^{(j/j\,-\,1)}\right)^2$
- $\hat{\omega}^{(j/j\,-\,1)} = 2\xi_v^{(j/j\,-\,1)}$

for $j = 1, \ldots, n$ are plugged into the equations.

To simplify the notation we define the following:

$$\mathcal{M}^*(t) = \frac{1}{2}M^*(R(u^{(j/j\,-\,1)}(t))), \quad j = 1, \ldots, n$$
$$\mathcal{N}^*(t) = \frac{1}{2}N^*(R(u^{(j/j\,-\,1)}(t)), \frac{1}{2}\hat{\xi}_v^{(j/j\,-\,1)}) \quad j = 1, \ldots, n$$

which simplifies eq.(3.4) to

$$\dot{\boldsymbol{\xi}}_v = \mathcal{M}^*(t)^{-1}(-\mathcal{N}^*(t)\boldsymbol{\xi}_v + F^*) \tag{3.5}$$

These equations evolve on the Lie Algebra of $\mathbb{S}^3$. But in order to integrate the system, we need accompanying information about the location on the the group as well. The equations 1.15 $\dot{u}^{(i/i\,-\,1)} = u^{(i/i\,-\,1)} \star \xi^{(i/i\,-\,1)}$ provides the

required information. They must be written as a matrix-vector expression. We use the notation:

$$\mathcal{Q} = \begin{bmatrix} Q(u^{(1)}) & & & \\ & Q(u^{(2/1)}) & & \\ & & \ddots & \\ & & & Q(u^{(n/n-1)}) \end{bmatrix}$$

which is a diagonal block-matrix with blocks of size 4×4, with the matrix representation 1.18 of the quaternion product on the diagonal. The time-derivative of the essential generalized coordinates represented as quaternions can then be written:

$$\begin{bmatrix} \dot{u}^{(1)} \\ \vdots \\ \dot{u}^{(n/n-1)} \end{bmatrix} = \mathcal{Q} \begin{bmatrix} \xi^{(1)} \\ \vdots \\ \xi^{(n/n-1)} \end{bmatrix} \tag{3.6}$$

The system is written as a vector-valued ODE by defining the $7 \cdot n$ term vector composed of the concatenated essential generalized coordinates and velocities. (The real part of the velocity quaternion is left out as it is always zero).

$$y = \begin{bmatrix} u^{(1)} \\ \vdots \\ u^{(n/n-1)} \\ \xi_v^{(1)} \\ \vdots \\ \xi_v^{(n/n-1)} \end{bmatrix}$$

Concatenating eq.(3.6) with eq.(3.5) yields the vector valued ODE:

$$\begin{bmatrix} \dot{q} \\ \dot{\boldsymbol{\xi}}_v \end{bmatrix} = \begin{bmatrix} \mathcal{Q}\xi \\ \mathcal{M}^*(t)^{-1}(-\mathcal{N}^*(t)\boldsymbol{\xi}_v + F^*) \end{bmatrix} \tag{3.7}$$

Thus we have the system expressed as a vector valued first-order differential equation.

$$\dot{y} = f(t, \underline{y}),$$

This new system made up of the equation 3.4 together with $\dot{u} = u \star \xi$ forms a system evolving on $(\mathbb{S}^3, T_I\mathbb{S}^3)$.

Next we discuss the inversion of the $\mathcal{M}^*$ matrix.

3.1.1 Solution to linear system

When evaluating 3.7 the equations eq.(3.4) must be solved. For a n-body system modeled with the direct representation of $SO(3)$, the M^* matrix will be of size $3n \times 3n$. As the terms of eq.(3.4) are generated during execution, we may evaluate M^*, N^* and F^* on each timestep t_j, and view the equations as a linear system of equations.

$$Ax = y, \quad A \in \mathbb{R}^{m \times m},\ x, y \in \mathbb{R}^m$$

which is then solved numerically for $\dot{\xi}_v$ at time t_j.

A first notion would be to apply Gauss elimination to the system. Unfortunately this algorithm can be unstable for problems with poor condition numbers, even with partial pivoting. Normally one would make an assessment of the condition of the problem before accepting a solution from a numerical solver. But during computation of the ODE with short timesteps, a Runge-Kutta solution scheme will call the ODE-function thousands of times, making the assessment on each and every call infeasible. But we are in luck, because such considerations are in fact not required at all!

By taking a closer look at $M^* = B^\top MB$, reveals a very useful property.

Theorem 3.1.1 (Positive definiteness of M^*)**.** *The matrix*

$$M^* = B^\top MB$$

is positive definite.

Proof. The mass matrix M 2.19 is positive definite by definition. The blocks on the diagonal of the form $M_{2j-1,2j-1} = m_j I$ are all diagonal matrices and thus symmetric positive definite. The blocks $M_{2j,2j} = J_j$ are the moment of inertia tensors which are always symmetric positive definite.

It is easily seen by inspection that the columns of the B-matrix are always linearly independent. The block-identity matrices located in a step wise manner in the B-matrix which do not align horizontally with the one another, as can be seen on the "spy" plot for the B-matrix for the system of length 4 displayed in 3.1.

So with $\forall y \neq 0$

$$y^\top B^\top MBy = x^\top Mx > 0$$

as $x \neq 0$ due to the linear independence of the columns of B. Since M is symmetric, $M^* = B^\top MB$ will be as well. This proves that M^* is positive definite. □

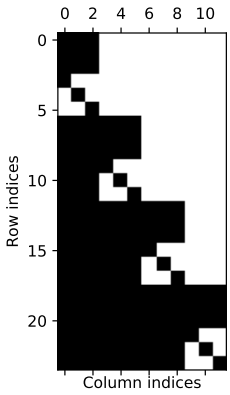


Figure 3.1: Spy plot of the B-matrix

The M^* matrix is small and dense in addition to being symmetric positive definite. The best algorithm for solving such systems of linear equations is the Cholesky decomposition which decompose symmetric positive definite matrices A to the form.

$$A = T^\top T$$

where T is a upper triangular matrix. [Tre97].

Applying the factorization we obtain $\mathcal{M}^* = T^\top T$ followed by a forward and a backward substitution procedure to the system, we can write:

$$\begin{bmatrix} \dot{q} \\ \dot{\boldsymbol{\xi}}_v \end{bmatrix} = \begin{bmatrix} \mathcal{Q}\xi \\ T^{-1}T^{\top -1}\left(-\mathcal{N}^*\boldsymbol{\xi}_v + F^*\right) \end{bmatrix}$$

Solution of a linear system by Cholesky decomposition is a backward stable algorithm in the sense that it minimizes rounding errors.This makes it preferable over the Gauss factorization which is categorized as "practically stable" [Tre97, p. 166]. The backward stable nature of the solution of the system by Cholesky factorization makes the solution to the sub-problem 3.4 as stable as it can possibly be on a computer with rounding errors.

Another advantage of the Cholesky decomposition is that the arithmetical cost is half of that required for Gauss elimination. This is a minor boost to computation time, although most of the computation effort is spent on constructing the B and $\dot{B}$ matrices.

However it should be noted that the error potential when using Gauss in this case is practically non-existent. Gauss elimination is very accurate

when applied to positive-definite systems. Normally partial pivoting is applied to ensure stability, which adds to the arithmetic cost. But partial pivoting is not required for symmetric positive systems, and therefore the relatively cheap Gauss elimination on pure form is well suited for this problem. Nevertheless, it can never beat the Cholesky factorization, which will be the method applied throughout the numerical simulations.

3.1.2 Choice of ODE integration scheme

The system of first order nonlinear ODE's 3.7 can readily be solved by a standard ODE integration scheme. To choose a scheme that is well suited to solving the multi-body system, the solution space must be taken into consideration.

The solution evolves on the unit-sphere $\mathcal{S}^3$ of quaternions of unit length. For the solution of the integration scheme to be accurate, it must preserve the unit-length of the quaternions with each step of the algorithm. This may at first seem like a problem which would require the application of a geometric integrator, but this is in fact not required. We observe that the length of a quaternion

$$|u(t)| = \sqrt{u^\dagger \star u}$$

is a quadratic quantity. To conserve this property of the solution 2.1 from [HLW05, Chapter 4, p. 101] provides precisely what we need.

Theorem 3.1.2 (Quadratic conservation law)**.** *Gauss-Legendre methods conserve quadratic polynomials.*

Thus we can conclude that all Gauss-Legendre methods integrates differential equations on the $\mathbb{S}^3$ manifold exactly, which makes them a prime candidate for solving the system. Consider the ν step implicit Runge-Kutta scheme:

$$\boldsymbol{k}_j = \boldsymbol{y}_n + h \sum_{i=1}^{\nu} a_{j,i} \boldsymbol{f}(t_n + c_i h, \boldsymbol{k}_i), \quad j = 1, 2, \ldots, \nu$$

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n h \sum_{i=1}^{\nu} b_j \boldsymbol{f}(t_n + c_j h, \boldsymbol{k}_j)$$

From [Ise09, Chapter 3] we have the methods three Gauss-Legendre methods: The implicit midpoint rule (IMR), 2-step Gauss-Legendre (GL2) and 3-step Gauss-Legendre (GL3). The parameters of the schemes are displayed in the Butcher tableaux below.

$$\begin{array}{c|c} \frac{1}{2} & \frac{1}{2} \\ \hline & 1 \end{array} \quad \text{(IMR)}$$

$$\begin{array}{c|cc} \frac{1}{2} - \frac{\sqrt{3}}{6} & \frac{1}{4} & \frac{1}{4} - \frac{\sqrt{3}}{6} \\ \frac{1}{2} + \frac{\sqrt{3}}{6} & \frac{1}{4} + \frac{\sqrt{3}}{6} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \quad \text{(GL2)}$$

$$\begin{array}{c|ccc} \frac{1}{2}-\frac{\sqrt{15}}{10} & \frac{5}{36} & \frac{2}{9}-\frac{\sqrt{15}}{15} & \frac{5}{36}-\frac{\sqrt{15}}{30} \\ \frac{1}{2} & \frac{5}{36}+\frac{\sqrt{15}}{24} & \frac{2}{9} & \frac{5}{36}-\frac{\sqrt{15}}{24} \\ \frac{1}{2}+\frac{\sqrt{15}}{10} & \frac{5}{36}+\frac{\sqrt{15}}{30} & \frac{2}{9}+\frac{\sqrt{15}}{15} & \frac{5}{36} \\ \hline & \frac{5}{18} & \frac{4}{9} & \frac{5}{18} \end{array} \tag{GL3}$$

Higher order methods are available, but the implementation costs exceeds the rewards in accuracy. While the three-step scheme has the highest order, the two-step Gauss-Legendre scheme is a better compromise between computational cost and accuracy and is therefore the preferred method for the rest of this thesis.

Runge-Kutta steps

The implicit schemes has a drawback in that the intermediate steps are a solution of a nonlinear system of equations. Solving the system requires additional computation which can be expensive in some cases. In this thesis, the nonlinear system is solved by a simple functional-iteration [Ise09, p. 123]. Given the steps of the method expressed as the nonlinear system:

$$\xi_j = y_n + h\sum_{i=1}^{\nu} a_{j,i} f(t_n + c_i, \xi_i)$$

This is a nonlinear system of the standard form:

$$\underline{\beta} = hg(\underline{\beta}) + \underline{\alpha}$$

This can be solved by a functional iteration scheme. Provided a initial guess, we have the iterative algorithm.

$$\underline{\beta}^{[i+1]} = hg(\underline{\beta}^{[i]}) + \underline{\alpha}$$

The initial guess is provided by a explicit Runge-Kutta scheme, with time-nodes coinciding with those of the implicit scheme. The iteration is repeated until the difference between two iterations falls below a specified tolerance $|\underline{\beta}^{[i+1]} - \underline{\beta}^{[i]}|_\infty < tol$.

A issue with the functional iteration approach is the rate of convergence. For stiff problems, the step-length h must be severely restricted in order to accurately solve for the steps. And thus, the functional iteration approach is

not a option for stiff ODE's as any advantage of the Implicit method is offset by the computation cost. But as will be seen in the following section, the functional-iteration is able to converge to the steps of the scheme in only a few iterations for systems without high accelerations. And the simple implementation makes it appealing.

Accuracy and Stability considerations

The implicit Gauss-Legendre methods conserve quadratic quantities provided that the steps of the method are solved exactly. And thus the accuracy of the scheme hinges on our ability to solve the non-linear system for the steps. For practical computations we must expect some errors from rounding errors. If the error grows too large, and the solution starts to meander away from unit length in the coordinates, a option is to periodically project the solution back onto the group of unit-quaternions. As mentioned before, a advantage of utilizing quaternions is the fact that this process simply involves normalizing the quaternion-coordinate. The projection process on the SO(3) group on the other hand involves the more costly process of orthogonolizing the matrix coordinate.

3.2 Analysis of numerical solutions

In this section the numerical solution to the coordinate free model is analyzed. To verify that the coordinate-free formulation that has been derived in the previous chapter models the behavior of a three-dimensional pendulum, we compare the simulations to those for the classical models for solid pendulums. A single and a double solid pendulum is modeled as a planar pendulum with parametrizing angles and simulated with standard explicit scheme to a strict error tolerance. The simulations are then compared to a coordinate-free formulation for the same system. The coordinate-free models must produce a solution identical to the traditional models, provided that the coordinate-free model is displaced the same angle about the same axis as the classical model, and that there are no external forces other than gravity.

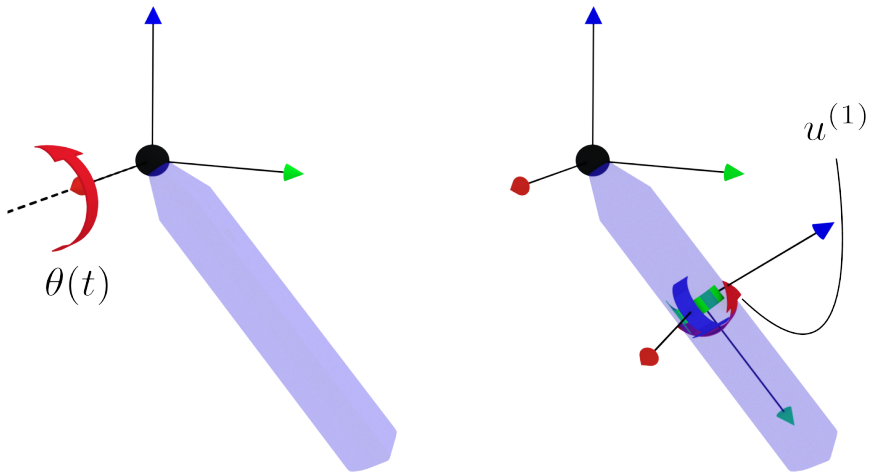
Another measure on the validity of the solution is the conservation of energy and unit-length. The solution to a conservative system should conserve the energy exactly as well as the unit-length of the quaternion coordinates. These quantities will be examined for a single body system and a four-body system.

The numerical solution of the models will in the beginning be computed with the three-step Gauss-Legendre scheme (GL3) to the a high accuracy to investigate the model, and the conservation of associated properties. Then finally we will examine methods of lower order with more reasonable computation costs, while maintaining accuracy to a acceptable degree.

3.2.1 Comparison to classical model

The simulation of a coordinate-free model of the solid three-dimensional pendulum is computed. The solution is then compared to the well known equations for a classical physical pendulum parametrized by one angle. The behavior of the classical pendulum is well known, and will be a reference solution to the coordinate-free model.

A pendulum shaped as a long narrow rod is fixed at point O located at one end and free to rotate. With center of mass located at $\frac{l}{2}$. The parameters of the pendulum body is: The center of mass is located at the volumetric center of the rod. The moment of inertia is taken to be that of a rectangular prism.



(a) Classical parametrized planar pendulum (b) Coordinate-free three-dimensional pendulum

Figure 3.2: Single solid pendulum

Model 1: Classical solid pendulum

The rotation of the single solid pendulum is parametrized by the angle $\theta(t)$ about the e_1 axis. (see fig.(3.2a)). The restoring torque on the pendulum is:

$$M = -mg\frac{l}{2}\sin(\theta)$$

The moment of inertia is $J = \frac{1}{3}ml^2$. Newton's second law for rotating rigid bodies provides the equations of motion $J\ddot{\theta} = M$. Inserting the restoring torque we obtain the classical model for the three-dimensional solid pendulum.

$$J\ddot{\theta} = -\frac{3g}{l}\sin\theta$$

In order to integrate the system numerically, the system is rewritten to a first-order form:

$$\dot{y}_1 = f_1(t, y_1), \quad y_1 = \begin{bmatrix}\theta(t)\\ \dot{\theta}(t)\end{bmatrix}$$

System Parameters	
Length l	2m
Width & Breadth	0.2m
Mass m	50kg

Table 3.1: Parameters for single three-dimensional pendulum

where

$$f_1(y_1) = \begin{bmatrix} \dot{\theta} \\ -mg\frac{l}{2}\sin\theta \end{bmatrix} \tag{3.8}$$

Model 2: The coordinate-free MFM equations

The coordinate free equations describes the solid three-dimensional pendulum fixed at one end, and free to rotate about all three axes (see fig.(3.2b)).

The mass and length of the pendulum is the same as the 1-dimensional system of equations. Provided that the three-dimensional pendulum starts from a initial configuration rotated the same angle θ as the 1-dimensional system about e_1 and both oscillates under the effect of gravity, the dynamical behavior of the two models must be identical.
The equations are assembled from the B, $\dot{B}$, D and M matrices. For the single three-dimensional joint these are trivial to formulate:

$$B = \begin{bmatrix} -R^{(1)}\hat{s}_{1,1} \\ I \end{bmatrix} \qquad \dot{B} = \begin{bmatrix} -R^{(1)}\hat{\omega}^{(1)}\hat{s}_{1,1} \\ 0 \end{bmatrix}$$

The mass matrix is according to the definition:

$$M = \begin{bmatrix} mI & 0 \\ 0 & J \end{bmatrix},$$

where J is the inertia tensor for a prism.
Lastly, the D-matrix is according to the definition 2.15:

$$D = \begin{bmatrix} 0 & 0 \\ 0 & \hat{\omega} \end{bmatrix}$$

Thus we have the system 3.7

$$\dot{y}_2 = f_2(t, y_2), \quad y_2 = \begin{bmatrix} u^{(1)}(t) \\ \xi_v^{(1)}(t) \end{bmatrix}$$

for the coordinate free model of the single three-dimensional pendulum.

Comparison of numerical simulations

The numerical simulation for 3.8 are computed with the `solve_ivp` ODE solver from the python scientific package `scipy.integrate` package with a global and relative error tolerance $rtol = 1e-13$ and $atol = 1e-13$. The chosen explicit scheme used by `solve_ivp` solver is the RK45 scheme. This solution is then regarded as a exact solution. Then3.7 is simulated with the GL3 scheme. The initial conditions for the two systems are:

$$y_1(t_0) = \begin{bmatrix} \theta(t_0) \\ \dot{\theta}(t_0) \end{bmatrix}, \text{ where } \begin{cases} \theta(t_0) = \theta_0 \\ \dot{\theta}(t_0) = 0 \end{cases} \tag{3.9}$$

$$y_2(t_0) = \begin{bmatrix} u^{(1)} \\ \xi_v^{(1)} \end{bmatrix}, \text{ where } \begin{cases} u_0^{(1)}(t_0) = \cos\left(\frac{\theta_0}{2}\right) \\ u_v^{(1)}(t_0) = \sin\left(\frac{\theta_0}{2}\right)\underline{e}_1 \\ \xi_v^{(1)}(t_0) = \underline{0} \end{cases} \tag{3.10}$$

Where the initial angle of displacement is: $\theta_0 = 0.1$. The system eq.(3.7) is integrated with the 3-step Gauss-Legendre scheme GL3.

Step-length To obtain a reference solution from the classical model, we determine the step-length necessary to obtain a solution with a global error below a required tolerance. By computing

$$y_1(t) = y_{exact}(t) + Ch^p$$
$$y_2(t) = y_{exact}(t) + C\left(\frac{h}{2}\right)^p$$

then subtracting $x_2(t_N)$ from $x_1(t_N)$ and solving for C, sufficient step-length is obtained by requiring that $h^p < \frac{1}{C}e_{tol}$. Choosing step lengths $h = 0.02$ and integrating over $T = [0, 1]$ with the 6'th order 3-step Gauss-Legendre method and requiring a tolerance of 10^{-14} gives the step length: $h = 0.04$. The step-length $h = 0.01$ is not too computationally demanding, while providing sufficient accuracy.

GL3 - Step Tolerance The non-linear solver applied to solve the steps of the GL3 scheme is set to iterate to a tolerance of $tol = 1e-16$ to obtain a accurate solution of the steps of the implicit scheme as possible.

Simulation Solving on the time interval $T = [0, 10]$ by discretizing $t_n = t_0 = 0 < t_1 < \cdots < t_N = T$ with time-step $h = 0.01$, the numerical solutions $y_1(t_n) = \begin{bmatrix} \theta(t_n) \\ \dot{\theta}(t_n) \end{bmatrix}$ and $y_2(t_n) = \Big[u(t_n), \xi_v(t_n)\Big]_n$ on the discrete grid are obtained. The solution to the coordinate free system must be mapped to Euler angles before the solutions can be compared. By inverting u_0 in the definition:

$$\begin{bmatrix} u_0 \\ u_v \end{bmatrix} = \begin{bmatrix} \cos(\theta/2) \\ \sin(\theta/2)\underline{n} \end{bmatrix}$$

we obtain the solution solution to the coordinate-free model in Euler angles:

$$\theta_2(t_n) = 2\arccos(u_0(t_n)) \tag{3.11}$$

The solutions of the coordinate-free and the classical models are compared by subtracting

$$e = \theta_1(t) - \theta_2(t).$$

The obtained error is displayed in fig.(3.3). The discrepancy at the end of

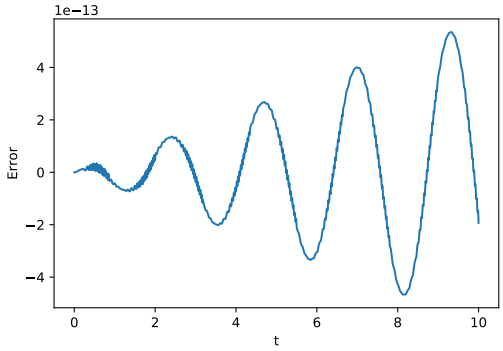


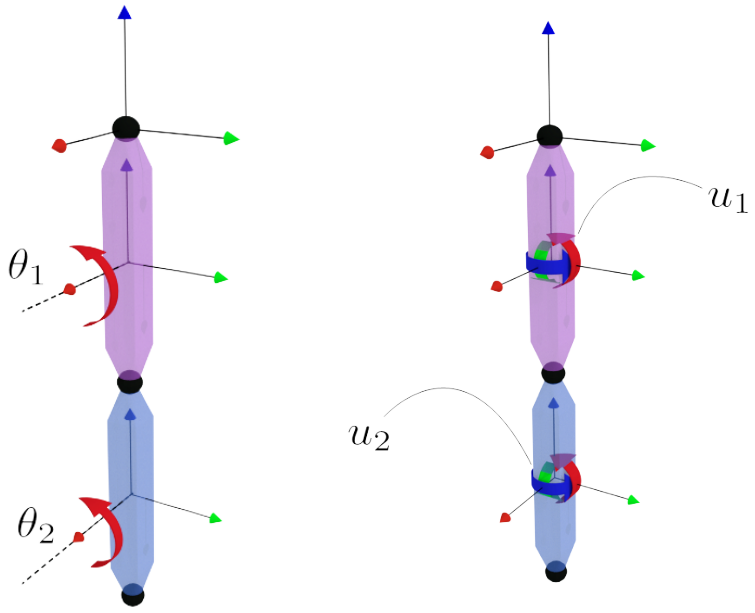
Figure 3.3: Difference in θ between reference solution and coordinate free solution solved with GL3 and h=0.01 for the single three-dimensional pendulum

the simulation is of the order of $1.92e{-}13$. The error at the end of simulation is at the same order as the absolute and relative tolerance of the explicit solver.

3.2.2 Double Pendulum Comparison

We verify the coordinate-free equations modelling the behavior of a multi-body system, by comparing them to the model constructed with the standard approach of the MFM, which applies generalized coordinates and symbolic expressions. Like in the previous section, the two models are solved numerically, the coordinate-free solution is converted to euler-angles and the two solutions are compared.

System-description The double pendulum is constructed from two rods idealized as two long rectangular prisms the first connected to a joint at origo, and the second connected at one end to a joint at the lower end of the first joint. The Local reference frames are located at the centers of mass which are taken to be the centers of the volumetric shape of the rectangular prisms, and the $\underline{e}^3$ axis of the frame is aligned along the longest side of the bodies. The systems are illustrated in fig.(3.4b) and fig.(3.4a). The



(a) Planar double pendulum (b) Coordinate-free model

Figure 3.4: Double three-dimensional pendulum

numerical values of the parameters for each body in the system are those given in 3.1. The moments of inertia are those for a rectangular prism.

The classical planar pendulum with parametrizing angles is derived with the moving frame method (MFM). Symbolic expressions for the B- and $\dot{B}$-matrices are obtained by rewriting the kinematic equations for the translational and rotational velocities. The resulting equations are ODE's in the parametrizing angles and their time-derivatives.

$$\dot{y}_1 = f_2(t, y_2), \quad y_1 = \begin{bmatrix} \theta_1(t) \\ \theta_2(t) \\ \dot{\theta}_1(t) \\ \dot{\theta}_2(t) \end{bmatrix}$$

The explicit derivation may be found in the appendix A.2.1.

The second coordinate-free model is as before 3.7 with length 2. The explicit expressions for the B and $\dot{B}$ matrices are shown in the appendix A.2.2. The matrices M and D are formulated according to the definitions eq.(2.19) and 2.15. The resulting equations are ODE's in the essential generalized coordinates and velocities with the unit quaternions as representation space.

$$\dot{y}_2 = f_2(t, y_2), \quad y_2 = \begin{bmatrix} u^{(1)}(t) \\ u^{(2/1)}(t) \\ \xi_v^{(1)}(t) \\ \xi_v^{(2/1)}(t) \end{bmatrix}$$

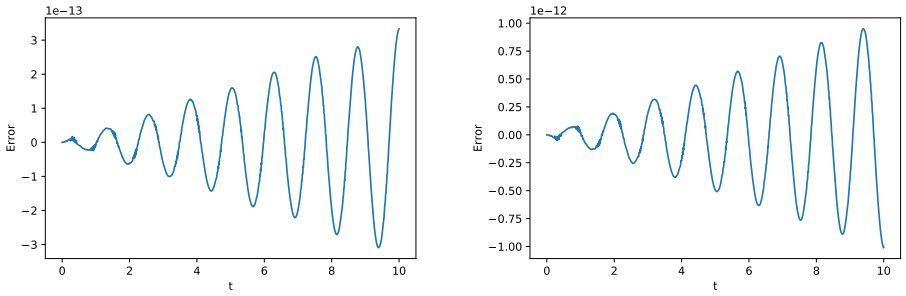
Comparison of numerical simulations

The initial conditions for the two systems are:

$$y_1(t_0) = \begin{bmatrix} \theta_1(t_0) \\ \theta_2(t_0) \\ \dot{\theta}_1(t_0) \\ \dot{\theta}_2(t_0) \end{bmatrix}, \text{ where } \begin{cases} \theta_1(t_0) = \theta_0 \\ \theta_2(t_0) = 0 \\ \dot{\theta}_1(t_0) = 0 \\ \dot{\theta}_2(t_0) = 0 \end{cases} \tag{3.12}$$

$$y_2(t_0) = \begin{bmatrix} u^{(1)}(t_0) \\ u^{(2/1)}(t_0) \\ \xi_v^{(1)}(t_0) \\ \xi_v^{(2/1)}(t_0) \end{bmatrix}, \text{ where } \begin{cases} u_1(t_0) = \begin{bmatrix} \cos\left(\frac{\theta_0}{2}\right) \\ \sin\left(\frac{\theta_0}{2}\right)\underline{e}_1 \end{bmatrix} \\ u_2(t_0) = I \\ \xi_1(t_0) = 0 \\ \xi_2(t_0) = 0 \end{cases} \tag{3.13}$$

Where the initial angle of displacement is: $\theta_0 = 0.1$. The numerical simulation for the model with parametrizing angles are computed with `scipy.integrate.solve_ivp` with the RK45 scheme with error tolerance $rtol = 1e{-}13$ and $atol = 1e{-}13$. The system 3.7 is integrated with the 3-step Gauss-Legendre scheme GL3 on $t = [0,\ 10]$. The Converting the coordinate free solution with 3.11 and comparing it to the solution of the traditional model yields the results displayed in fig.(3.5). The solution of the two models remain close (error less than



(a) Body 1 (b) Body 2

Figure 3.5: Difference in θ between reference solution and coordinate free solution solved with GL3 and h=0.01

$1e - 12$) throughout the simulation. The double pendulum is the smallest system which is also chaotic, meaning two initial values close to each other can produce widely different solutions. We see however little of this phenomenon, which can be attributed to the fact that the angular displacement in the initial condition was small.

3.2.3 Conservation of unit length and energy

A indication on how well the integration scheme is able to accurately solve the system is its ability to maintain invariants of the system. In this section we will investigate the conservation of quaternion unit length and total energy of the system. The solutions are computed with the three-step Gauss-Legendre method to the strictest possible tolerance. When we are assured that the properties of the system are conserved, we discuss lowering the order of the method and loosening the tolerance on the solution of the Runge-Kutta steps.

Unit length To evaluate the ability of the scheme 2.20 to maintain unit length of the generalized coordinates, the length of the quaternion coordinate is computed at each discrete time node. Then computing the error as follows

$$e_l = |q(t_n)| - 1$$

System energy The system 3.7 for a friction-less pendulum is conservative. The error in energy is:

$$E = E_0 - \frac{1}{2}\dot{q}^\top M^* \dot{q} + mge_3^\top \underline{x}_{cm} \tag{3.14}$$

The relative energy conservation error is computed as:

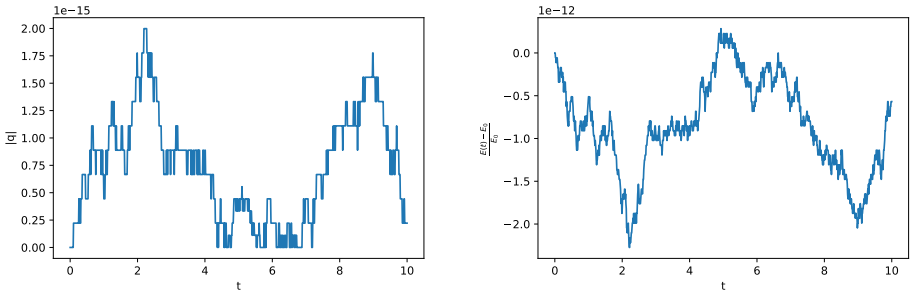
$$e = \frac{E(t) - E_0}{E_0} \tag{3.15}$$

Conservation for single spherical pendulum

The conservation of energy and unit length is computed for the solution of the coordinate free model of the single pendulum from the previous section, and the results are displayed in fig.(3.6).

The deviation in unit-length remains steady at about $1e-15$ which is at about machine accuracy. The conservation of unit length is very good, and there is hardly any need for re-normalizing the quaternion coordinates.

The computed deviation from the initial energy-sum remains at the level of 10^{-11}. While the accuracy is not as good as the conservation of unit length, this accuracy is still very good. The order of the method is 6, and we can expect a error of $1e-7$ for a step-length of $h = 0.01$, thus the energy is conserved to the same order as the method.



(a) Deviation from unit length (b) Relative energy conservation error

Figure 3.6: Property conservation for the Single spherical pendulum simulated with GL3

Conservation for 4-body spherical pendulum

To study the behavior of a multi-link spherical solid pendulum a system of length 4 is analyzed. Like before, local frames of reference are assigned to

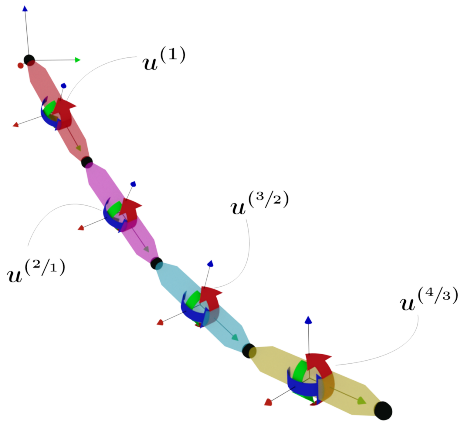


Figure 3.7: 4-body solid spherical pendulum

each body in the system, located at the center of mass and aligned with the principal axes of the moment of inertia.

With the moving frame - method, extending the system with a arbitrary number of additional joints is trivial. The B- and $\dot{B}$-matrices are constructed iteratively from the general expression for their block-entries from 2.3 and 2.4. Then constructing the mass matrix according to eq.(2.19) and D-matrix

according to eq.(2.15), and all the components are in place to formulate eq.(2.20).

To investigate numerical properties of the multi-body three-dimensional pendulum of length 4. The resulting B- and $\dot{B}$ are 24×12 matrices.

The resulting first-order system of differential equations eq.(3.7) is a 28-term vector ODE-system.

$$y(t_n) = \begin{bmatrix} u^{(1)}(t_n) \\ \vdots \\ u^{(4/3)}(t_n) \\ \xi_v(t_n) \\ \vdots \\ \xi_v(t_n) \end{bmatrix}_n$$

The initial conditions for the simulation is:

$$\begin{aligned} u^{(1)}(t_0) &= \begin{bmatrix} \cos\left(\theta_0/2\right) \\ \sin\left(\theta_0/2\right) \cdot \underline{e}_1 \end{bmatrix} \\ u^{(2)}(t_0) &= u^{(3)}(t_0) = u^{(4)}(t_0) = I \\ \xi_v^{(1)}(t_0) &= \xi_v^{(2/1)}(t_0) = \xi_v^{(3/2)}(t_0) = \xi_v^{(4/3)}(t_0) = \underline{0} \end{aligned}$$

where $\theta_0 = 0.1$

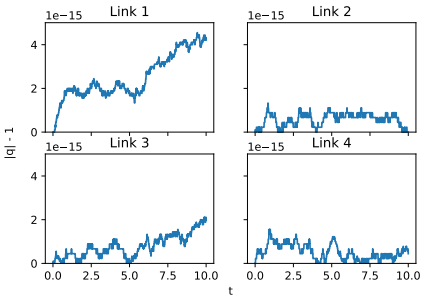
The system eq.(3.7) is integrated on the discrete time steps $t_0 = 0 < t_1 < \cdots < t_N = T$ of the time span $t = [0, 10]$ and with a time step $h = 0.01$. The numerical integration scheme is the three-step Gauss-Legendre scheme GL3. The computed numerical solution $y(t_n)$ is obtained.

Conservation of unit length and energy

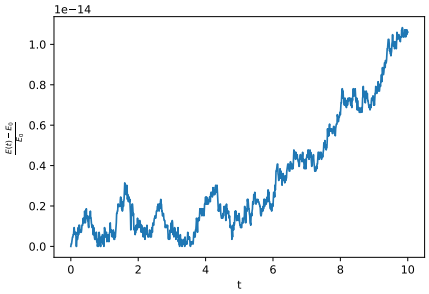
Again, we investigate how well the scheme maintains the invariants of the system. The energy and unit-length-conservation is computed in the same manner as the previous section on the single pendulum. The unit lengths of the quaternions are on the order of ϵ_{mach} throughout the time-span. The energy conservation error remains at the order of ϵ_{mach} throughout the timespan. Results are displayed in fig.(3.8).

3.2.4 Tolerance of solution to Runge-Kutta steps

In the previous sections, the solutions were computed with the three-step Gauss-Legendre methods. The tolerance of the non-linear solver used to



(a) Conservation of unit length - 4 body pendulum



(b) Energy Conservation -

Figure 3.8: Property conservation, 4 body pendulum simulated by GL3 scheme with strict step-tolerance

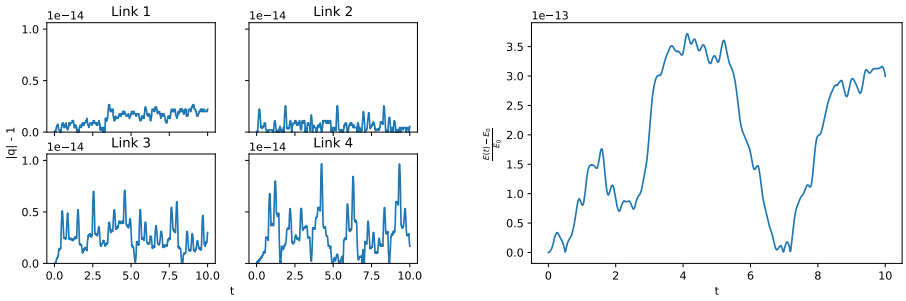
solve for the steps of the implicit method was set to 1e-15. This is excessive since the order of the solution scheme is 6, thus it makes sense to solve the steps of the method to a tolerance which corresponds to the order of the method itself. In addition, the computation of the steps of the three-step Gauss-Legendre method is costly which leads us to consider the GL2 scheme.

The proper tolerance on the non-linear solver implemented to determine the intermediate steps of the Gauss-Legendre is obtained by comparing experiments on the the four-body three-dimensional pendulum from section 3.2.3. The system is solved with explicit RK45 solver computed with a absolute- and relative- tolerance of $10^{-}13$. The system is solved with the implicit scheme GL2 with a tolerance $tol = 1e - 9$ on the solution of the intermediate steps: Simulating the system we see that the explicit and implicit solver has the following error in unit length at $T = 10$: The GL2

Error in quaternion unit-length at time T and function evaluations						
	Link 1	Link 2	Link 3	Link 4	Relative energy error	Function evaluations
RK45	5.55e-15	2.47e-14	2.24e-14	2.50e-14	7.22e-14	15794
GL2	2.22e-15	5.55e-16	3.00e-15	1.66e-15	3.00e-13	11334

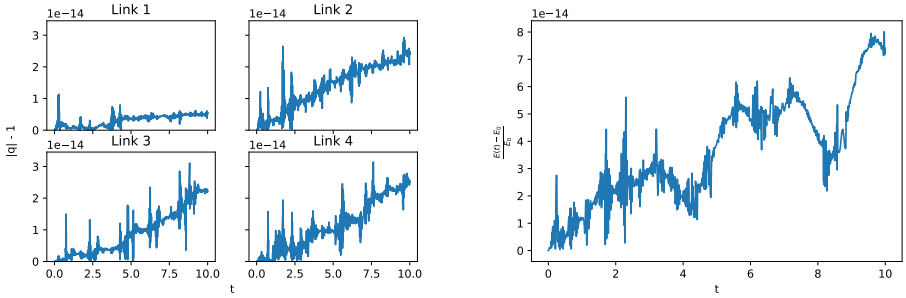
Table 3.2: Error in quaternion coordinate unit-length at end of integration

method with the intermediate steps of the scheme computed to a tolerance $tol = 1e - 9$ computes the solution to accuracy comparable to the explicit scheme set to the strictest tolerance.



(a) Deviation from unit length (b) Energy Conservation

Figure 3.9: Property conservation, 4 body pendulum simulated by GL2 scheme with looser step-tolerance



(a) Deviation from unit length (b) Energy conservation

Figure 3.10: Property conservation, 4 body pendulum simulated by RK45 scheme with strict absolute and relative error-tolerance

The advantage of the Gauss-Legendre method comes to light when observing the number of calls to the ODE-function. The explicit scheme calls the function a total number of 15794 times, while the two-step implicit Gauss-Legendre function calls the function a total number of 11334 times. Thus even when the implicit method must solve a non-linear system for each step of integration, it is still able to compute the system to a accuracy rivaling the explicit method at a significantly lower number of iterations.

We also observe that stepping down to a 2-step Gauss-Legendre method, and reducing the tolerance of the solution of the steps does not have a severe impact to the accuracy of the solution. In fact, while the error in conserved energy is at the order of 10^{-13} observed the plot fig.(3.9), the error at the

end of simulation is at the order of 10^{-15}, a error one may consider exact. Thus we may safely use this lower order method provided a exceptional accuracy is not required.

3.2.5 Computation time for N-body pendulum

We here investigate the computation time for systems of increasing number of bodies in the n-body pendulum-model solved with the three-step Runge-Kutta method. The links of the N-body pendulum individually have the parameters 3.1. The pendulum starts from rest with the first body displace $0.1rad$ about the x-axis. The system is integrated on $t = [0, 10]$ with a time step $h = 0.01$. The maximum RAM allocated by the python-process solving the equations was 90MB. The average computation times of 7 runs are displayed in 3.3.

Timings	
Number of links in system	Computation time
4	$719ms \pm 4.31ms$
8	$2.27sec \pm 15.8ms$
16	$10.7sec \pm 122ms$
32	$45.6sec \pm 1.55s$
48	$1min43s \pm 1.94s$
64	$3min24s \pm 5.52s$

Table 3.3: Computation times for incresing system length computed with the GL3 scheme

Plotting the time and links reveals more regarding the increases in computation time. The loglog plot of system of time and system length is linear with a slope ≈ 2. This shows that the computation time increases quadratically for a increase in system length, thereby verifying the observation in 2.5 that the expressions composing the B- and $\dot{B}$-matrix grows quadratically. In addition this also indicates that the computation time is dominated by the generation of the B- and $\dot{B}$-matrices.

The maximum amount of RAM allocated to the python process during execution of the solver was 90MB. In practical terms, the computation time is more constraining than the memory requirements.

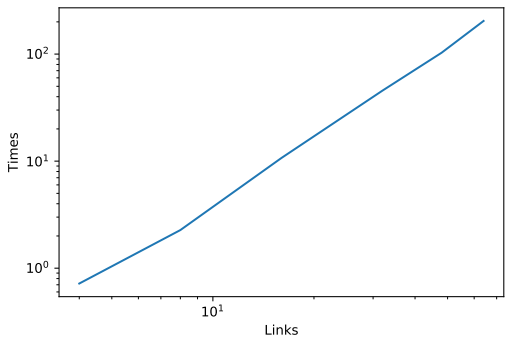


Figure 3.11: Loglog plot of system length and computation time for systems of increasing length

3.3 Applications

In this section we will look at some application of the coordinate-free formulation of the Euler-Lagrange equations. First we simulate a solution of the heavy top.

A strength of the MFM method is the ease with which external forces may be included into the model. A number of models with spring-damper systems and torsional spring-dampers applied to them are simulated. With these simulations we seek to demonstrate the versatility and robustness of the method in combination with solution schemes.

For these models, visual animations have been rendered to better give a better impression of the systems under consideration. The videos are uploaded to youtube, and the links to the animations are given at the end of each section.

3.3.1 Heavy symmetrical top

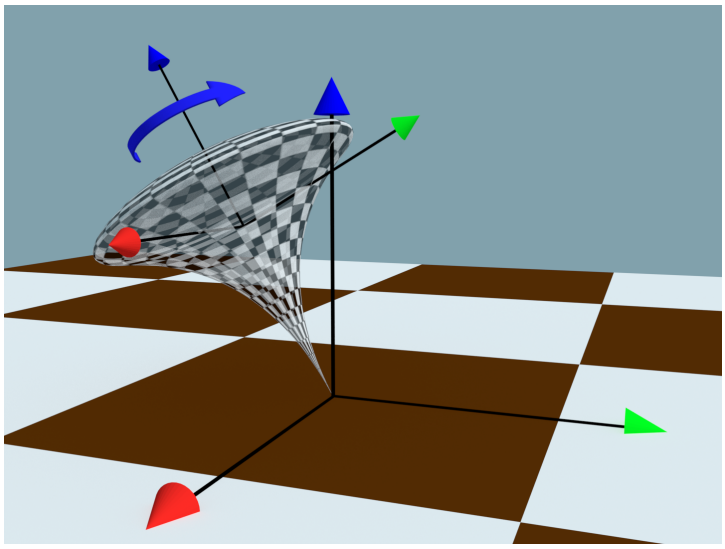


Figure 3.12: Symmetric heavy top

We will now simulate a solution to a heavy symmetrical top, with one point fixed. The top is idealized as a rod with a flat disc mounted at the top. The radius and length of the disc and rod are:

The center of mass is the average of the two composed bodies. The density of the rod and the disc is $1000^{kg}/_{m^3}$. The moment of inertia is moments

System Parameters		
	disc	rod
radius	$0.4m$	$0.05m$
h	$0.05m$	$0.4m$

Table 3.4: Parameters for the solid top

of inertia of the two bodies, translated with the parallel axis theorem.

Simulation The generalized coordinates of the top is $u \in \mathbb{S}^3$, and the resulting system 3.7 is a 7-term vector-valued ODE.

The initial conditions are:

$$y(t_0) = \begin{bmatrix} u(t_0) \\ \xi_v(t_0) \end{bmatrix} \text{ where } \begin{cases} u(t_0) = \begin{bmatrix} \cos(\frac{\theta(t_0)}{2}) \\ \sin(\frac{\theta(t_0)}{2})\underline{e}_1 \end{bmatrix} & \theta(t_0) = \frac{\pi}{6} \\ \xi_v(t_0) = \dot{\theta}(t_0)\underline{e}_3 & \dot{\theta}(t_0) = 10. \end{cases}$$

where the initial angle $\theta(t_0)$ is a angular displacement of the figure axis away from the vertical position, and $\dot{\theta}(t_0)$ is the initial angular velocity of the top about its local axis.

The system eq.(3.7) is integrated on the discrete time steps $t_0 = 0 < t_1 < \cdots < t_N = T$ with the time range $[0, 10]$ and time step $h = 0.01$. The numerical integration scheme is the three-step Gauss-Legendre scheme GL3. The computed numerical solution $y(t_n)$ is obtained. A animation of the solution is displayed in.

Animation of heavy symmetric top

https://youtu.be/cOL68IbniPE

For the symmetric top, the validity can be verified by inspecting the momentum conservation of the system. A well known fact regarding the behavior of the heavy top is conservation of momentum about the figure-axis. A accurate computation of the system must conserve the angular momentum. The conservation of this property can be observed by projecting the momentum onto the figure axis (third axis).

$$e = H_3(t) - H_3(t_0) = {e_3^{(1)}}^\top J\underline{\omega}(t) - {e_3^{(1)}}^\top J\underline{\omega}(t_0)$$

The numerical computation reveals that the momentum is conserved exactly.

3.3.2 4-body pendulum with torsional springs

An interesting modification to the model is the introduction of torsional springs at the joints which resist bending. Considering a 4-body chain as displayed in 3.13 where the parameters are given in 3.1. On the joint between body j and body $j+1$, we have three torsional springs attached. The springs are attached at the joints of the 4-body pendulum such that the neutral configuration of the system is the vertical orientation of all the links. The resisting moments are illustrated in 3.13 with the round arrows located at the joints. The orientation of body $j+1$ can be converted to euler angles

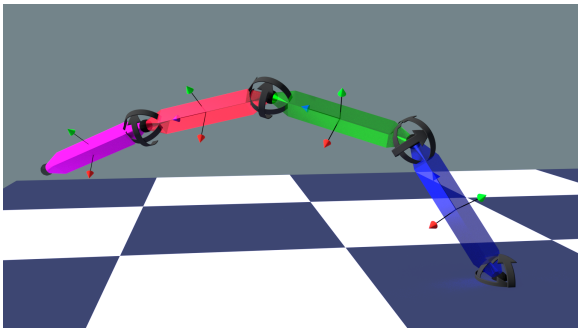


Figure 3.13: 4-body pendulum with torsional springs

$(\theta(t),\ \phi(t),\ \psi(t))$ about the local axes of the frame (j). The orientation is converted to euler-angles by computing the rotation matrix generated by the 3-2-1 sequence of Euler angles

$$R^{(j/j-1)} = R_z(t)R_y(t)R_x(t)$$

$$= \begin{bmatrix} \cos(\theta)\cos(\phi) & & \\ \cos(\theta)\sin(\psi) & & \\ -\sin(\theta) & \sin(\phi)\cos(\theta) & \cos(\phi)\cos(\theta) \end{bmatrix}$$

Solving the entries in the first collumn and last row, yields the expressions for the euler angles in terms of the rotation matrix (4 terms in the upper

right are not used and left out).

$$\theta(t) = \arcsin(-R_{3,1}^{(j/j-1)})$$
$$\phi(t) = \arctan(\frac{R_{3,3}^{(j/j-1)}}{R_{3,2}^{(j/j-1)}})$$
$$\psi(t) = \arctan(\frac{R_{2,1}^{(j/j-1)}}{R_{1,1}^{(j/j-1)}})$$

If we interpret the Euler angles as rotational displacement away from the neutral position of the body, the moment applied to body j is:

$$\boldsymbol{e}^{(j)}M = -\underline{e}^{(j-1)}k \begin{bmatrix} \theta \\ \phi \\ \psi \end{bmatrix}$$

And the moment applied to $j+1$

$$\boldsymbol{e}^{(j+1)}M = \boldsymbol{e}^{(j+1)}R^{(j/j-1)^\top}k \begin{bmatrix} \theta \\ \phi \\ \psi \end{bmatrix}$$

The moments are introduced into the force vector M in 2.16. The spring coefficients are given in 3.5

Spring coefficients	
k_1	$4000\frac{kg \cdot m}{s^2}$
k_2	$3000\frac{kg \cdot m}{s^2}$
k_3	$2000\frac{kg \cdot m}{s^2}$
k_4	$1000\frac{kg \cdot m}{s^2}$

Table 3.5: Coefficients for torsional springs about the local e_1 and e_2 axes

Simulation Initially the bodies are rotated a angle θ_0 about the first axis, followed by a rotation about the second axis. The orientations are computed by the quaternion product:

$$u^{(1)}(t_0) = u^{(2/1)}(t_0) = q^{3/2)}(t_0) = q^{4/3)}(t_0) = \begin{bmatrix} \cos(\frac{\theta_0}{2}) \\ \sin(\frac{\theta_0}{2})\underline{e}_1 \end{bmatrix} \star \begin{bmatrix} \cos(\frac{\theta_0}{2}) \\ \sin(\frac{\theta_0}{2})\underline{e}_2 \end{bmatrix}$$

The system starts from rest.

$$\xi_v^{(1)}(t_0) = \xi_v^{(2/1)}(t_0) = \xi_v^{(3/2)}(t_0) = \xi_v^{(4/3)}(t_0) = \underline{0}$$

The system eq.(3.7) is integrated on the discrete time steps $t_0 = 0 < t_1 < \cdots < t_N = T$ with the time range $[0, 15]$ and time step $h = 0.01$. The numerical integration scheme is the three-step Gauss-Legendre scheme GL2. The computed numerical solution $y(t_n)$ is obtained. The conservation of quaternion-unit length is displayed in fig.(3.14). We observe that dispite

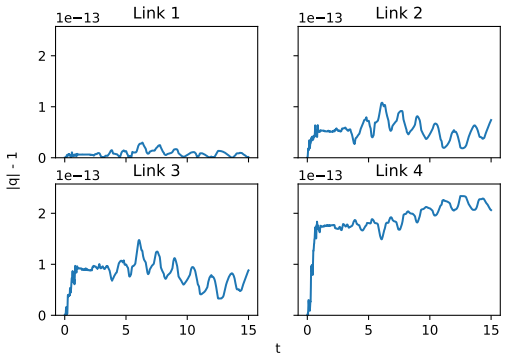


Figure 3.14: Conservation of quaternion-unit lengths

large displacements, the scheme is able to conserve unit-length without the need for normalization.

Animation of the system solution is displayed in the video in the following link.

Animation of 4-body pendulum with torsional springs

https://youtu.be/EA6zu04TB7I

3.3.3 16-body pendulum

As an example of a significantly longer system. We define a 16-body system. The first link is hinged to origo, and every subsequent body is hinged at the end of the previous link. The hinges are all spherical, allowing the links to rotate in any direction. The parameters of the bodies are those given in 3.1. The dynamic system 3.7 is a 122-term vector ODE system generating a solution on $(\mathbb{S}^3)^{16}$ and $T_I\mathbb{S}^3$.

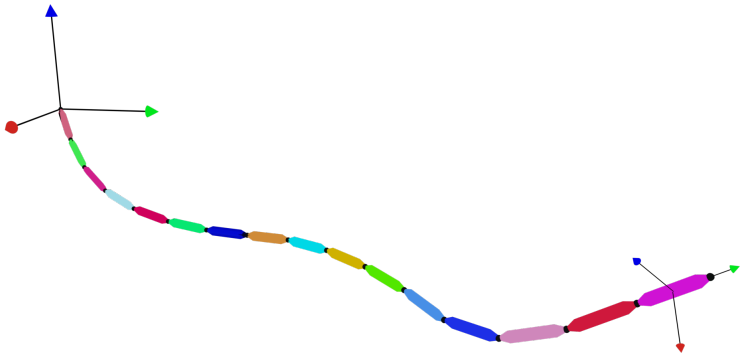
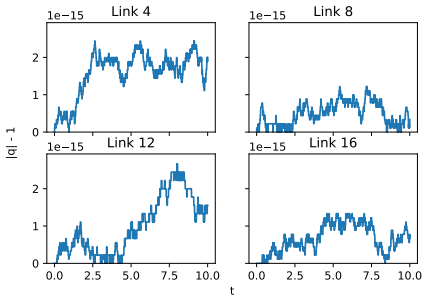


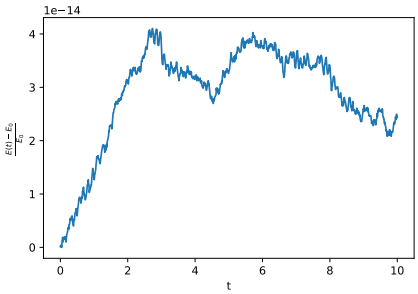
Figure 3.15: 16-body pendulum

Energy-/Length- conservation

To assess the accuracy, we investigate the conservation of length of quaternion-coordinates $e = |q(t_n)| - 1$ and energy according to 3.14 and 3.15. The initial condition for the first coordinate $q_1(t_0)$ is expressed by 1.13 with $\theta_0 = \frac{\pi}{8}$ and the remaining coordinates without any displacement. The results from integration with the GL2 method are displayed in 3.16a and 3.16b. Unit length



(a) Conservation of unit length for quaternion coordinates on link 4, 8, 12 and 16



(b) Energy conservation

Figure 3.16: Conservation of energy and quaternion-unit length for the 16-body pendulum solved with GL2 scheme with a step length h=0.01

and energy is conserved even for this long system.

With external forces

Several types of external forces may be applied to the model. This spring-damper is added to the system. The spring-damper with neutral length l_N is attached at the end of the last body in the system at one end and in the inertial frame at the other. This is illustrated in fig.(3.17). The attachment

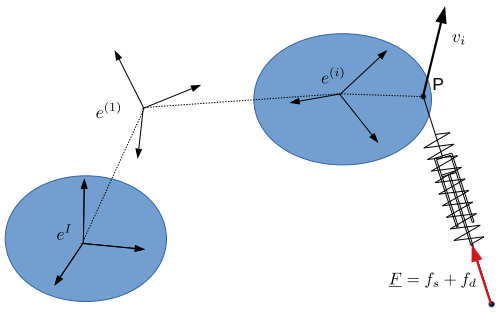


Figure 3.17: Spring loaded/damped system

point on the last body of the 16-body pendulum is.

$$(\boldsymbol{e}^I, \underline{x}_p(t)) = \overline{\boldsymbol{e}}^I E^{(16)}(t) \begin{bmatrix} I & \underline{s}_p \\ 0 & 1 \end{bmatrix}$$

and the attachment point in the inertial frame is

$$(\boldsymbol{e}^I, \underline{x}_{cp}) = \overline{\boldsymbol{e}}^I \begin{bmatrix} I & \underline{x}_{cp} \\ 0 & 1 \end{bmatrix}$$

The vector from the connection points in the inertial frame is then:

$$\underline{x}_s(t) = \underline{x}_p^I(t) - \underline{x}_{cp}^{(I)}$$

from which we compute the spring stretching/compression:

$$s = l_N - \left|\underline{x}_s(t)\right|$$

and direction of the spring

$$\underline{n}(t) = \frac{\underline{x}_s(t)}{\left|\underline{x}_s(t)\right|}$$

Thus the spring-forces in the inertial frame is:

$$\underline{f}_s^I = \boldsymbol{e}^I k \cdot s \cdot \underline{n}(t)$$

The moment applied to the body is:

$$M_s(t) = \underline{s}_p \times \underline{f}_s^{(16)}(t), \quad \underline{f}_s^{(16)}(t) = {R^{(16)}}^\top \underline{f}_s^I(t)$$

where the index denotes that the vectors are expressed in the local frames.

Damper

The velocity of the moving point on the 16'th body where the spring/damper is connected is

$$\overline{\boldsymbol{e}}^I \dot{E}^{(16)} = \overline{\boldsymbol{e}}^I E^{(16)} \Omega^{(16)} \begin{bmatrix} I & \underline{x}_p \\ 0 & 0 \end{bmatrix}$$

The velocity in the direction of the damper is

$$\underline{v}_d(t) = \underline{n}(t) \cdot \underline{v}_p(t)$$

And the damper force defined as proportional with the velocity with which the damper is extended is then.

$$\underline{f}_d^I = c \cdot \underline{v}_d(t)$$

Moments is computed in the same manner as the spring force.

$$M = \underline{s}_p \times \underline{f}_d^{(16)}, \quad \underline{f}_d^{(16)}(t) = {R^{(4)}}^\top \underline{f}_d^I$$

Assembling the two, the force vector is:

$$F = \begin{bmatrix} \boldsymbol{e}^I - g \\ 0 \\ \vdots \\ \boldsymbol{e}^I(-g + \underline{f}_d(t) + \underline{f}_s(t)) \\ \boldsymbol{e}^{(16)}(\underline{M}_d(t) + \underline{M}_s(t)) \end{bmatrix}$$

Torsional damper

A issue with a large multi-body system is the chaotic nature of the differential equations modeling the system. After a short time-span, high frequency oscillations arise. This causes severe issues for the non-linear solver solving the steps of the implicit scheme, which is unable to converge for severe oscillations. To mitigate this problem, dampers on the joints between bodies is introduced. The dampers apply a moment to the pair of bodies are computed as:

$$M_d^{(j)} = -c\underline{\omega}^{(j/j\,-\,1)}$$

and

$$M_d^{(j-1)} = cR^{(j/j\,-\,1)}M_d^{(j)}$$

Other Parameters Some more parameters regarding the system.

Spring coefficient, k	$800^{kg}/_{s^2}$
Damper coefficient, c	$50^{kg}/_{s}$
Joint-damper coefficient	$70^{kg\,\cdot\,m^2}/_{s}$

Table 3.6: Torsional spring & damper parameters

The moments of inertia is idealized as a rectangular prism with length $2m$ and sides $1/4m$.

Simulation The resulting first-order equations eq.(3.7) is a 112-term vector ODE-system in terms of the vector:

$$y(t_n) = \begin{bmatrix} u^{1)}(t_n) \\ \vdots \\ u^{(16/15)}(t_n) \\ \xi_v^{(1)}(t_n) \\ \vdots \\ \xi_v^{(16/15)}(t_n) \end{bmatrix}_n$$

The quaternions and angular velocities in the initial initial conditions $\underline{y}(t_0)$ are:

$$u^{(1)}(t_0) = \begin{bmatrix} \cos\left(\theta_0/2\right) \\ \sin\left(\theta_0/2\right) \cdot e_1 \end{bmatrix}$$
$$u^{(2/1)}(t_0) = \cdots = u^{(16/15)}(t_0) = I$$
$$\xi^{(1)}(t_0) = \cdots = \xi^{(16/15)}(t_0) = \underline{0}$$

The system eq.(3.7) is integrated on the discrete time steps $t_0 = 0 < t_1 < \cdots < t_N = T$ with the time range $[0, 10]$ and time step $h = 0.01$. The numerical integration scheme is the three-step Gauss-Legendre scheme GL3. The computed numerical solution $y(t_n)$ is obtained. A animation of the solution is displayed in the following link:

Animation of 16-body spring loaded pendulum

https://youtu.be/a_ibo_unOog

The length conservation of link 4, 8, 12 and 16 are displayed in fig.(3.18). We see that the unit-length of the quaternion solution is well maintained.

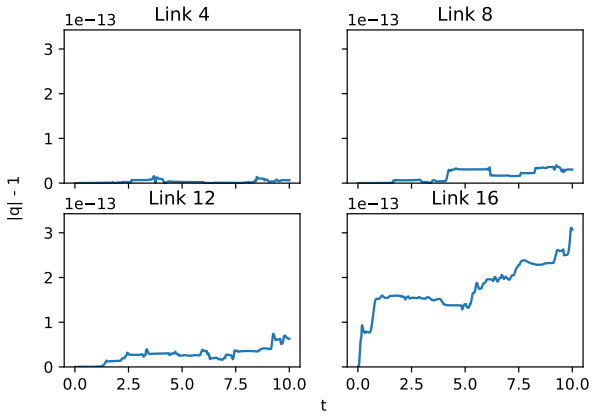


Figure 3.18: Conservation of quaternion length for 16-body system

3.3.4 The 64-body pendulum

As an example of a significantly longer system. We define a 64-body system. The first link is hinged to origo, and every subsequent body is hinged at the

Figure 3.19: 64-body pendulum

end of the previous link and free to rotate in any direction. The parameters of the bodies are those given in 3.1. The dynamic system 3.7 is a 672-term vector ODE system generating a solution on $(\mathbb{S})^{16}$. The pendulum is coiled in the x/y-plane and drops down upon release. To suppress wild oscillations, torsional dampers are attached at the joints. Parameters of the bodies in the system are given in 3.1, the moments of inertia is the one for a rectangular cuboid and the joint-damper coefficients are $70^{kg}/_{s}$.

Animation of 64-body pendulum falling from rest

https://youtu.be/efhnb3jQkas

3.4 Remarks on simulations

In this chapter we have shown a method of rewriting the system of equations to standard form for ODE's by utilizing the positive definiteness of the M^* matrix. It has been shown that the implicit Gauss-Legendre Runge-Kutta schemes are solve the system accurately for a computation cost which is lower than for explicit Runge-Kutta schemes. In particular, the two-step Gauss-Legendre scheme has been demonstrated to provide a suitable compromise between accuracy and computation cost.

Computation cost We have shown that the computation increases quadratically for a increase in system length. The RAM required to evaluate the ODE function was negligible, thus verifying that the generation of the B- and $\dot{B}$-matrices numerically upon evaluation from the expressions derived in chapter 2 is far more efficient than deriving the numerical expressions for the matrices on beforehand.

Chapter 4

Conclusion

In this thesis we have derived a coordinate-free formulation of the Euler-Lagrange equations within the formalism of the Moving-Frame method. The fusion of the coordinate-free approach by Leok et.al. [TLM17] and the Moving Frame Method by Murakami et.al [MI19] has been shown to produce the equations of motion for multibody systems which are stable, cheap and which can be applied to n-body solid spherical pendulums of arbitrary length.

Expressions for generating the B- and $\dot{B}$-matrices have been derived. It has been shown that these expressions are efficient for computing the matrices numerically when required from the evaluation of the ODE function; both in terms of computation and memory cost. The proposed method has also turned out to be easily scalable to larger multibody systems. In implementation terms, an advantageous feature of this method is that the characteristics of the systems are entirely contained within the B and $\dot{B}$-matrices. When a scheme for computing the B-matrix is established, a standard framework for computing the equations can then be employed to solve the equations. By experiment it has been shown that computations of system is feasible at least up to length $n = 64$. Longer systems are tractable although computation time increases significantly.

The implicit Gauss-Legendre Runge-Kutta schemes have been shown to be accurate and efficient for computing numerical solutions to the coordinate-free differential equations. The implementation of the methods is fairly accurate even when the implicit iterations (solved by fixed point iteration) are solved with a fairly low tolerance. This is much better than an explicit RK-scheme which needs to be computed with much stricter error-tolerance in order to obtain the same level of accuracy. It has also been shown that the inversion of the M^*-matrix , due to its symmetric positive definiteness, is

very stable. This property also rewards us with savings in computation time in comparison to other schemes and effectively eliminates worries regarding the inversion process.

The application of the group of unit quaternions as generalized coordinates was shown to be a good choice for generalized coordinates of rotating bodies described by Lagrangian mechanics. They are cheap in terms of storage, and the simulation of the resulting equations of motion by the implicit Gauss-Legendre schemes maintains automatically the solution within the group of unit quaternions, a property that does not hold for explicit RK-schemes.

In this thesis we considered only free spherical rotations. In many applications, rotations may be restricted to one or two axes, or the body restricted to translate along a line or on a plane. We believe it is possible to describe the generating expressions for the corresponding constrained B and $\dot{B}$ matrices, however, this was outside the scope of this thesis and is a topic for further work.

Possible further work A possible future development is to derive methods for constraining the system. With the application of Lagrangian multipliers, the motion of the system may be further constrained, opening the way to many additional applications in engineering.

Another possible avenue of study is to recast the problem as the Hamiltonian, and study the application of symplectic methods to the system. By applying the Legendre transform to the Lagrangian and working out the terms we arrive at the Hamiltonian:

$$H = \frac{1}{2} B^\top M^{*-1} B$$

Even though this has a nice compact form, it is difficult to analyze and even more difficult to compute. In particular Hamilton's equations involve the partial derivative of the Hamiltonian with respect to the generalized coordinates. As M^* is dependent on the generalized coordinates, the partial derivative of its inverse needs to be evaluated which is not a simple undertaking.

Appendix A

Appendix

A.1 Rewriting the operator

The right adjoint operator may be rewritten as follows:

$$Ad_{\exp(\hat{\omega}t)}\hat{\xi} = \widehat{R^{\top}(t)\xi}$$

The new expression for the adjoint is obtained with the identity:

$$Ad_{\exp(\hat{\omega}t)}\hat{\xi} = \exp(ad_{t\hat{\omega}})\hat{\xi}$$

The new form of the adjoint is obtained by rewriting the exponential of the little adjoint. We have $R = \mathrm{e}^{\hat{\omega}t}$.

$$\begin{aligned}
Ad_{\exp(\hat{\omega}t)}\hat{\xi} &= \exp(ad_{t\hat{\omega}})\hat{\xi} \\
&= \hat{\xi} + [\hat{\xi},\hat{\omega}]t + \frac{1}{2!}[[\hat{\xi},\hat{\omega}],\hat{\omega}]t^2 + \frac{1}{3!}[[[\hat{\xi},\hat{\omega}],\hat{\omega}],\hat{\omega}]t^3 + \mathcal{O}(t^4) \\
&= \hat{\xi} + \widehat{\hat{\xi}\omega}t + \frac{1}{2!}[\widehat{\hat{\xi}\omega},\hat{\omega}]t^2 + \frac{1}{3!}[[\widehat{\hat{\xi}\omega},\hat{\omega}],\hat{\omega}]t^3 + \mathcal{O}(t^4) \\
&= \hat{\xi} - \widehat{\hat{\omega}\xi}t + \frac{1}{2!}[-\widehat{\hat{\omega}\xi},\hat{\omega}]t^2 + \frac{1}{3!}[[-\widehat{\hat{\omega}\xi},\hat{\omega}],\hat{\omega}]t^3 + \mathcal{O}(t^4) \\
&= \hat{\xi} - \widehat{\hat{\omega}\xi}t + \frac{1}{2!}\widehat{\hat{\omega}^2\xi}t^2 - \frac{1}{3!}\widehat{\hat{\omega}^3\xi}t^3 + \mathcal{O}(t^4) \\
&= \widehat{\mathrm{e}^{-\hat{\omega}t}\xi} \\
&= \widehat{R^{\top}(t)\xi}
\end{aligned}$$

A.2 Kinematics for the double pendulum

A.2.1 B and $\dot{B}$ -matrices with paramterizing coordinates

The symbolic expression for the B-matrix is most easily tackled head-on by explicitly developing the expressions for the translation velocities in the inertial frame $\dot{x}_1,\ \dot{x}_2$ and the angular velocities in the local frame $\underline{\omega}_1, \underline{\omega}_2$ using the fundamental rotation matrices with symbolic trigonometric expressions in the entries.

$$\begin{aligned}
\dot{x}_1(t) &= R^{(1)}\hat{\omega}^{(1)}\underline{s}_{1,1} \\
\dot{x}_2(t) &= \tfrac{\mathrm{d}}{\mathrm{d}t}x_1(t) + R^{(1)}\hat{\omega}^{(1)}\underline{s}_{1,2} + R^{(2)}\hat{\omega}^{(2)}\underline{s}_{2,1}
\end{aligned}$$

The matrices are the fundamental rotation matrices about the x-axis; $R^{(1)}(t) = \mathrm{e}^{\hat{\boldsymbol{e}}_1\theta(t)}$, $R^{(2/1)} = \mathrm{e}^{\hat{\boldsymbol{e}}_1\theta_2(t)}$ and the angular velocities are the Lie algebra accompanying these rotation matrices $\hat{\omega}^{(1)} = \mathrm{e}^{\hat{\boldsymbol{e}}\dot{\theta}_1(t)}$, $\hat{\omega}^{(2/1)} = \hat{\boldsymbol{e}}_1\theta_2(t)$. By writing out the expressions explicitly we obtain:

$$\underline{\dot{x}}_1(t) = \begin{bmatrix} 0 \\ -\frac{1}{2}l\sin\left(\theta_1(t)\right)\frac{d}{dt}\theta_1(t) \\ \frac{1}{2}l\cos\left(\theta_1(t)\right)\frac{d}{dt}\theta_1(t) \end{bmatrix}$$

$$\underline{\omega}^{(1)}(t) = \begin{bmatrix} \frac{\mathrm{d}}{\mathrm{d}t}\theta_1(t) \\ 0 \\ 0 \end{bmatrix}$$

$$\underline{\dot{x}}_2(t) = \begin{bmatrix} 0 \\ -\frac{1}{2}l\sin\left(\theta_1(t)+\theta_2(t)\right)\frac{d}{dt}\theta_2(t) + \left(-\frac{1}{2}l\sin\left(\theta_1(t)+\theta_2(t)\right) - l\sin\left(\theta_1(t)\right)\right)\frac{d}{dt}\theta_1(t) \\ \frac{1}{2}l\cos\left(\theta_1(t)+\theta_2(t)\right)\frac{d}{dt}\theta_2(t) + \left(\frac{1}{2}l\cos\left(\theta_1(t)+\theta_2(t)\right) + l\cos\left(\theta_1(t)\right)\right)\frac{d}{dt}\theta_1(t) \end{bmatrix}$$

$$\underline{\omega}^{(2)} = \begin{bmatrix} \frac{d}{dt}\theta_1(t) + \frac{d}{dt}\theta_2(t) \\ 0 \\ 0 \end{bmatrix}$$

Then by manually reordering the terms as a matrix vector expression in the euler angles $\dot{\theta}_1, \dot{\theta}_2$, the expressions are assembled into the B-matrix, and we obtain:

$$B = \begin{bmatrix} 0 & 0 \\ -\frac{l\sin\left(\theta_1(t)\right)}{2} & 0 \\ \frac{l\cos\left(\theta_1(t)\right)}{2} & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -\frac{l\sin\left(\theta_1(t)+\theta_2(t)\right)}{2} - l\sin\left(\theta_1(t)\right) & -\frac{l\sin\left(\theta_1(t)+\theta_2(t)\right)}{2} \\ \frac{l\cos\left(\theta_1(t)+\theta_2(t)\right)}{2} + l\cos\left(\theta_1(t)\right) & \frac{l\cos\left(\theta_1(t)+\theta_2(t)\right)}{2} \\ 1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

With the B-matrix formulated symbolically, the $\dot{B}$-matrix is obtained by simply taking the derivative of the B-matrix entry by entry. The M-matrix 2.19 and D-matrix 2.15 are formulated according to the definitions.

With all components of the system 2.20 defined we solve the system by introducing the angular velocity variables $\alpha_i = \alpha_i$, and rewriting the equations we obtain:

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \alpha_1 \\ \alpha_2 \end{bmatrix} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ M^{*-1}\left(-N^*\dot{q} + F^*\right) \end{bmatrix}$$

Where the inverse of the M^* matrix may be solved with the formula for the inverse of a 2×2 matrix. When solving the equations in terms of parametrizing coordinates, we need not be concerned for the conservation of any properties concerning the coordinates themselves, as was necessary for the quaternion coordinates. A explicit method such as the RK45 scheme is suitable for this problem.

A.2.2 Coordinate-free equations for the 2-body pendulum

B and $\dot{B}$ matrices are for the coordinate-free model are composed by the expressions eq.(2.30) and eq.(2.35). For the 2-body pendulum, these are as follows:

$$B = \begin{bmatrix} -R^{(1)}\hat{s}_{1,1} & 0 \\ I & 0 \\ B_1^{(1)} - R^{(1)}\hat{s}_{1,2}R^{(1)^\top} - R^{(2)}\hat{s}_{2,1}R^{(2/1)^\top} & -R^{(2)}\hat{s}_{2,1} \\ R^{(2/1)^\top} & I \end{bmatrix}$$

$$\dot{B} = \begin{bmatrix} -R^{(1)}\hat{\omega}^{(1)}\hat{s}_{1,1} & 0 \\ 0 & 0 \\ \dot{B}_{21}^{(1)} - R^{(1)}\hat{\omega}^{(1)}\hat{s}_{1,2} - R^{(2)}\left(\hat{\omega}^{(2)}\hat{s}_{2,1} - \hat{s}_{2,1}\hat{\omega}^{(2/1)}\right)R^{(2/1)^\top} & -R^{(2)}\hat{\omega}^{(2)}\hat{s}_{2,1} \\ -\hat{\omega}^{(2/1)}R^{(2/1)^\top} & 0 \end{bmatrix}$$

Bibliography

[Axl15] Sheldon Axler. *Linear algebra done right*. Springer, 2015.

[CR99] Evangelos A. Coutsias and Louis Romero. The quaternions with an application to rigid body dynamics. 1999.

[Gol00] Herbert Goldstein. *Classical Mechanics*. Addison Wesley, 2000.

[HJ85] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

[HLW05] Ernst Hairar, Christian Lubich, and Gerhard Wanner. *Geometric Numerical Integration*. Springer, 2005.

[IMKNZ00] Arieh Iserles, Hans Z. Munthe-Kaas, Syvert P. Nørsett, and Antonella Zanna. Lie-group methods. *Acta Numerica*, pages 216–365, 2000.

[Ise09] Arieh Iserles. *First Course in the Numerical Analysis of Differential Equations*. Cambridge University Press, 2009.

[Lac07] Claude Lacoursière. *Ghosts and Machines: Regularized Variational Methods for Interactive Simulations of Multibodies with Dry Frictional Contacts*. PhD thesis, Umeaa University, June 2007.

[MI19] Hidenori Murakami and Thomas J. Impelluso. *Moving Frame Method in Dynamics - A Geometrical Approach*. Pearson Publishing, 2019.

[MLS94] Richard M. Murray, Zexiang Li, and S. Shankar Sastry. *A Mathematical Introduction to Robotic Manipulation*. 1994.

[Ryk18] Thorstein Rykkje. Lie groups and the principle of virtual work applied to systems of linked rigid bodies, June 2018.

[Sti08] John Stillwell. *Naive Lie Theory*. Springer, 2008.

[Str15] Steven H. Strogatz. *Nonlinear Dynamics and Chaos*. Westview press, 2015.

[TLM17] Lee Taeyoung, Melvin Leok, and N. Harris McClamroch. *Global Formulations of Lagrangian and Hamiltonian Dynamics on Manifolds*. springer, 2017.

[Tre97] Nicholas Trefethen. *Numerical Linear Algebra*. SIAM, 1997.