DHM2020
L. Hanson et al. (Eds.)

doi:10.3233/ATDE200012

Defining Kinematic Chains for Musculoskeletal Optimal Control Simulations via Automatic Differentiation

Johann PENNER [a,1], Sigrid LEYENDECKER [a]
[a] *Chair of Applied Dynamics, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany*

Abstract. Many digital human model applications are based on optimal control simulations of the musculoskeletal system. These simulations usually involve the derivatives of the underlying kinematic and dynamic model, which are in general not easy to derive analytically. In the direct transcription method DMOCC, we use the discrete Euler-Lagrange equations together with a discrete null space matrix and a nodal reparametrization, which are embedded into a constrained optimization problem. The abstract and formalizable structure of this method offers many possibilities for automation. Therefore, we use the CasADi nonlinear optimization and algorithmic differentiation tool to automatically derive the discrete Euler-Lagrange equation and a valid discrete null space matrix. This allows us an efficient and easy implementation of the DMOCC method for large multibody systems.

Keywords. musculoskeletal optimal control simulations, discrete mechanics, automatic differentiation

1. Introduction

The representation of musculoskeletal systems with multibody dynamics and cable-like muscles has become an essential tool to analyse human movement [1]. The actuation of these models via Hill muscles requires the calculation of the muscle paths, their lengths and changes in length during the movement, which is determined by the muscle wrapping problem [2,3]. In order to model and solve complex human motion tasks, an appealing approach is to formulate optimal control problems and to approximate their solution numerically. The development of automatic differentiation in resent years reveals efficient ways to create and solve these optimal control problems, which we make use of in this paper. We thereby focus on an automatic generation of the discrete Euler-Lagrange equations and the discrete null space matrix, see [4–6].

When simulating the dynamics of musculoskeletal systems, we formulate the dynamics of multibody system representing bones and joints and the dynamic muscle paths [2, 3, 7, 8], which wrap smoothly over adjacent obstacles, in terms of discrete Lagrange mechanics [4–6]. Setting up new musculoskeletal models in the present formula-

[1]Corresponding Author: Johann Penner, Chair of Applied Dynamics, Friedrich-Alexander-Universität Erlangen-Nürnberg, Immerwahrstrasse 1, 91058 Erlangen, Germany; E-mail: johann.penner@fau.de.

tion is an abstract, formalizable process that offers a wide range of options for automation. In particular for open kinematic chains, we use a discrete null space matrix and a nodal reparametrization of the unknowns (redundant director based coordinates) via incremental generalized coordinates in order to reduce the dimension of the system size in two steps [5, 6].

Note that the discrete null space matrix is not unique and can be found in different ways [5, 6]. In fact, a particularly elegant way is to represent the discrete null space matrix as the Jacobian of the nodal reparametrization, which describes the kinematics of the multibody system. This Jacobian can be calculated using automatic differentiation. Consequently, since higher order derivative information is needed anyway for the optimisation tasks, this offers opportunities for simplifying and automating the model creation.

Efficient methods for the implementation of the discrete Euler–Lagrange equations for generic mechanical systems with tree structured, rigid multibody system are shown in [9–11]. The benefits of a derivation of the underlying system dynamic with automatic differentiation has for example been shown in [12]. The idea of this work is to generate the discrete Euler-Lagrange equations, the discrete null space matrix and all higher order derivatives for the direct transcription method discrete mechanics and optimal control for constrained systems (DMOCC) [13] with the help of automatic differentiation. This significantly shortens and simplifies model creation compared to manual implementations. The interaction of the skeleton with the muscles and the contact formulation is currently formulated in detail in another paper.

CasADi [14] is an open-source tool for nonlinear optimization and algorithmic differentiation that provides an easy framework for the implementation of the DMOCC method. Here, the director formulation allows us a very modular and automated design of discrete Euler-Lagrange equations, once the discrete Lagrangian of the system and a nodal reparametrisation are setup. The derivatives of these functions, i.e. the variational integrator, can easily be nested into the nonlinear optimization program with CasADi.

2. Method

The main objective of the simulations in this work is the control of a musculoskeletal system, which must be steered from a given initial state to a predefined final state. Therefore, we apply the DMOCC formulation [13] to a biomechanical system, where bones and joints are represented as multibody system [5, 6] and the muscle path is modelled as the shortest connection between two points on obstacle surfaces [2, 3]. As a result of that, the infinite dimensional optimal control problem is transcribed into a finite dimensional nonlinear programming problem that can be solved by any suitable standard algorithm.

2.1. Musculoskeletal Optimal Control Problem

To solve the optimization problem, we require higher order derivatives of the underlying dynamical system. The implementation these derivatives by hand is generally not an easy task. In addition, derivations are required for the discrete Euler-Lagrange equations and the discrete null space matrix. Consequently, the automation of all derivatives brings a great advantage for the creation and solution of optimal control problems.

2.1.1. Constrained Optimization Problem

The DMOCC [13] method can be summarized as

$$\min_{\boldsymbol{x}_d} J_d(\boldsymbol{x}_d)$$

$$\text{subject to} \quad \begin{array}{l} \cdot \text{ discrete Euler-Lagrange equations} \\ \cdot \text{ boundary and path conditions} \end{array}$$

where $\boldsymbol{x}_d = \{\boldsymbol{x}_n\}_{n=0}^N$ is the discrete optimisation vector with $N \in \mathbb{N}$, J_d is the discrete objective function, that is to be minimized subject to the discrete Euler-Lagrange equations of the system and further boundary and path conditions.

2.1.2. Discrete Euler-Lagrange Equations of the Musculoskeletal System

In our model, the discrete path $\boldsymbol{q}_d = \{\boldsymbol{q}_n\}_{n=0}^N$ is an approximation of the continuous path on a discrete time grid with constant time step $\Delta_t \in \mathbb{R}$ and $N+1$ time nodes. The discrete muscle path vector $\boldsymbol{\gamma}_d = \{\{\boldsymbol{\gamma}_{k,n}\}_{k=0}^K\}_{n=0}^N$ is defined on a discrete arc length grid with fixed arc length fraction $\Delta_s \in \mathbb{R}$ with $K \in \mathbb{N}$ and $K+1$ nodes. Note that, we have to solve for all $K-1$ unknown muscle path configurations $\boldsymbol{\gamma}_{k,n}$ at every time node.

We further choose the midpoint quadrature and finite differences to specify the discrete Lagrangian [4] for the mechanical system L_d and the geodesic L_γ. In addition, we use the discrete null space matrix $\boldsymbol{P}_d(\boldsymbol{q}_n)$ and the nodal reparametrisation $\boldsymbol{q}_{n+1} = \boldsymbol{F}_d(\boldsymbol{u}_{n+1}, \boldsymbol{q}_n)$ in terms of discrete local coordinate $\boldsymbol{u}_n$, see [5, 6]. The surface constraint function $\phi_d(\boldsymbol{q}_n, \boldsymbol{\gamma}_k)$ and the integrals of the scalar product of the constraints and the Lagrange multipliers $\mu_{k,n}$ are approximated in a similar way. This leads to the following discrete Euler-Lagrange equations

$$\boldsymbol{P}_d(\boldsymbol{q}_n)^T \cdot \left[\frac{\partial L_d(\boldsymbol{q}_{n-1}, \boldsymbol{q}_n)}{\partial \boldsymbol{q}_n} + \frac{\partial L_d(\boldsymbol{q}_n, \boldsymbol{q}_{n+1})}{\partial \boldsymbol{q}_n} - \frac{\partial \phi_d(\boldsymbol{\gamma}_k, \boldsymbol{q}_n)}{\partial \boldsymbol{q}_n}^T \cdot \mu_{k,n} \right] = 0 \tag{2}$$

$$\frac{\partial L_\gamma(\boldsymbol{\gamma}_{k-1}, \boldsymbol{\gamma}_k)}{\partial \boldsymbol{\gamma}_k} + \frac{\partial L_\gamma(\boldsymbol{\gamma}_k, \boldsymbol{\gamma}_{k+1})}{\partial \boldsymbol{\gamma}_k} - \frac{\partial \phi_d(\boldsymbol{\gamma}_k, \boldsymbol{q}_n)}{\partial \boldsymbol{\gamma}_k}^T \cdot \mu_{k,n} = 0 \tag{3}$$

$$\phi_d(\boldsymbol{\gamma}_k, \boldsymbol{q}_n) = 0$$

for $0 < n < N$ and $0 < k < K$, see [7, 8].

2.2. Automatic Derivation of the System Equations

The first step in creating a multibody simulation model, is the definition of the number, shape and connection of the rigid bodies. Knowing the dimensions, weight and location of every rigid body, one can easily define the Lagrangian of the system. The coupling of rigid bodies is achieved via configuration constraints that form a kinematic chain. The augmentation of the Lagrangian and discretization of Hamilton's principle then leads to time stepping equations that inherit certain characteristic properties of the continuous solution, the discrete Euler-Lagrange equations. To reduce the dimension of the resulting equations, it is possible to apply a discrete null space matrix and a nodal reparametrization [4–6].

This is an abstract and formalizable process that offers a wide range of options for automation. The main tasks here are the derivation of the Euler-Lagrange equations and the derivation of a null space matrix. Within this work, we use MATLAB for the implementation of the examples. Thereby, our formulation supports the null space method with a user defined reparametrization for oben kinematics chains and is internally based on CasADi [14].

2.2.1. Derivation of the Euler-Lagrange Equations

Lets consider a multibody system consisting of a kinematic chain with b rigid bodies. This yields a $12b$-dimensional configuration variable

$$\boldsymbol{q}(t) = \begin{bmatrix} \boldsymbol{q}^1 \\ \vdots \\ \boldsymbol{q}^b \end{bmatrix} \in \mathbb{R}^{12b}. \tag{4}$$

with $\boldsymbol{q}^\beta$, $\beta = 1,\dots,b$ representing the redundant coordinates of a rigid bodies. The Lagrangian of the system

$$\mathscr{L}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \mathscr{T}(\boldsymbol{q},\dot{\boldsymbol{q}}) - \mathscr{V}(\boldsymbol{q}) \tag{5}$$

is composed by the kinetic energy $\mathscr{T}(\boldsymbol{q},\dot{\boldsymbol{q}})$ and potential energy $\mathscr{V}(\boldsymbol{q})$, where $\dot{\boldsymbol{q}}(t) \in \mathbb{R}^{12b}$ is the velocity of the system. Approximating the integral over the Lagrangian in the interval $[t_n, t_{n+1}]$ via midpoint quadrature and finite differences, yield the discrete Lagrangian

$$L_d(\boldsymbol{q}_n, \boldsymbol{q}_{n+1}) = \Delta_t \mathscr{L}\left(\left(\frac{\boldsymbol{q}_{n+1}+\boldsymbol{q}_n}{2}\right), \left(\frac{\boldsymbol{q}_{n+1}-\boldsymbol{q}_n}{\Delta_t}\right)\right) \tag{6}$$

where $\boldsymbol{q}_n$ is approximating $\boldsymbol{q}(t_n)$. Whit the mass matrix $\boldsymbol{M}$ and the gravity vector $\boldsymbol{g}$ of right dimension, the first derivatives of the discrete Lagrangian contribute to the discrete Euler-Lagrange equations and hence the variational integrator. A pseudo code in a syntax similar to MATLAB can be read as

```
function [L_d] = d_Lagrangian(q_n,q_np1)
    q = (q_np1+q_n)/2;
    q_dot = (q_n-q_np1)/dt;
    L_d = dt*(q_dot'*M*q_dot/2 - q'*M*g);
end

import casadi.*
q_np1 = MX.sym('qnp1',number_of_bodies*12,1);
q_n = MX.sym('qn',number_of_bodies*12,1);
q_nm1 = MX.sym('qnm1',number_of_bodies*12,1);

d_EulerLagrange = casadi.Function('DEL',{q_nm1,q_n,q_np1}, ...
    {gradient(d_Lagrangian(q_n,q_np1),q_n) + ...
    gradient(d_Lagrangian(q_nm1,q_n),q_n)});
```

The discrete Euler-Lagrange equations `d_EulerLagrange(q_nm1,q_n,q_np1)` are a function of three consecutive configurations, which can easily be nested to the constrained optimization problem. Thereby, CasADi provides the functionality to take further higher-order derivatives of the discrete Euler-Lagrange equations and to generate C code from them.

2.2.2. *Derivation of the Null Space Matrix for Open Kinematic Chains*

Lets now assume that the b rigid bodies are connected via $b-1$ joints and one anchor, such that the incremental update function reads

$$\boldsymbol{q}_{n+1} = \boldsymbol{F}_d(\boldsymbol{u}_{n+1}, \boldsymbol{q}_n) = \begin{bmatrix} \boldsymbol{F}_d^1(\boldsymbol{u}_{n+1}, \boldsymbol{q}_n) \\ \vdots \\ \boldsymbol{F}_d^b(\boldsymbol{u}_{n+1}, \boldsymbol{q}_n) \end{bmatrix} \in \mathbb{R}^{12b} \tag{7}$$

with F_d^α, $\alpha = 1, \ldots, b$. The entries of the incremental coordinate $\boldsymbol{u}_n$ represent the degrees of freedom of all kinematic pairs in the chain. The total degrees of freedom DoF define the dimension of the discrete local coordinate, i.e. $\boldsymbol{u}_n \in \mathbb{R}^{DoF}$. The first rigid body of every kinetic chain is assumed to be either anchored or free in space. In this setup, the Jacobian of the nodal reparametrization

$$\boldsymbol{P}_d(\boldsymbol{q}_n) = \left. \frac{\partial \boldsymbol{F}_d(\boldsymbol{u}_{n+1}, \boldsymbol{q}_n)}{\partial \boldsymbol{u}_{n+1}} \right|_{\boldsymbol{u}_{n+1}=\boldsymbol{0}} \tag{8}$$

evaluated at $\boldsymbol{q}_n$ is a suitable null space matrix. A pseudo code can be read as

```
function [F_d] = n_Reparam(u_np1,q_n)
    F_d = [F_1(u_np1,q_n);...
           ...
           F_b(u_np1,q_n)];
end

import casadi.*
u_np1 = MX.sym('unp1',DoF,1);
q_n = MX.sym('qn',number_of_bodies*12,1);

d_NullSpace = casadi.Function('P',{u_np1,q_n}, ...
    {jacobian(n_Reparam(u_np1,q_n),u_np1)});
```

Thereby, the properties of the reparametrization are to be defined by the user. This gives our implementation a maximum of flexibility, while standard connections can be provided by a library. The function `d_NullSpace(u_np1,q_n)` is calculated in a preprocessing step and is hand over to the optimization problem.

3. Results

We show two examples of musculoskeletal optimal control simulations. While the first example serves as validation with simplified complexity, a model of the kinematic model

of the upper body is presented in the second example. As mentioned before, we use the direct transcription method DMOCC to transcribe the optimal control problem into a nonlinear program that adopts the properties of structure preserving integrators. The constrained optimisation problem is solved via IPOPT [15]. Alternately, `fmincon` or to search for global optima `multistart` [16] from MATLAB can be used.

3.1. Two Muscle Arm Model

In this example (Figure 1), the multibody system consists of two rigid bodies and a revolute joint, which represent the upper and lower arm connected by the elbow joint. The upper arm is fixed in space, the configuration of the lower arm is given as $\boldsymbol{q} = [\boldsymbol{x}_s, \boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_3]^T \in \mathbb{R}^{12}$, $\boldsymbol{n} \in \mathbb{R}^3$ is the joint axis and $\boldsymbol{\rho} \in \mathbb{R}^3$ is the joint location. The centre of mass is given by the vector $\boldsymbol{x}_s(t) \in \mathbb{R}^3$ and the orientation is indicated by an orthonormal body frame $\boldsymbol{d}_I(t) \in \mathbb{R}^3$ with $I = 1,2,3$ spanning a rotation matrix. In this example, the dynamics are discretized with $N = 50$ time nodes and the muscle path is discretized with $K = 20$ nodes per muscles. Furthermore, the nodal reparametrization in terms of the rotational degree of freedom $\boldsymbol{\theta}_n = u_n \boldsymbol{n}$ is given as

$$\boldsymbol{F}_d(u_{n+1}, \boldsymbol{q}_n) = \begin{bmatrix} \boldsymbol{x}_s - \mathrm{cay}(\widehat{\boldsymbol{\theta}_{n+1}}) \cdot \boldsymbol{\rho}_n \\ \mathrm{cay}(\widehat{\boldsymbol{\theta}_{n+1}}) \cdot \boldsymbol{d}_{I_n} \end{bmatrix} \tag{9}$$

where we use the Cayley map $\mathrm{cay} : so(3) \to SO(3)$. Herein $\widehat{(\bullet)} \in so(3)$ denotes a skew-symmetric matrix. We use the Cayley map because the differentiation of the Rodrigues formula for the evaluation of the exponential map does not work easily with automatic differentiation. The model comprise two muscles (biceps, triceps) and the actuation is

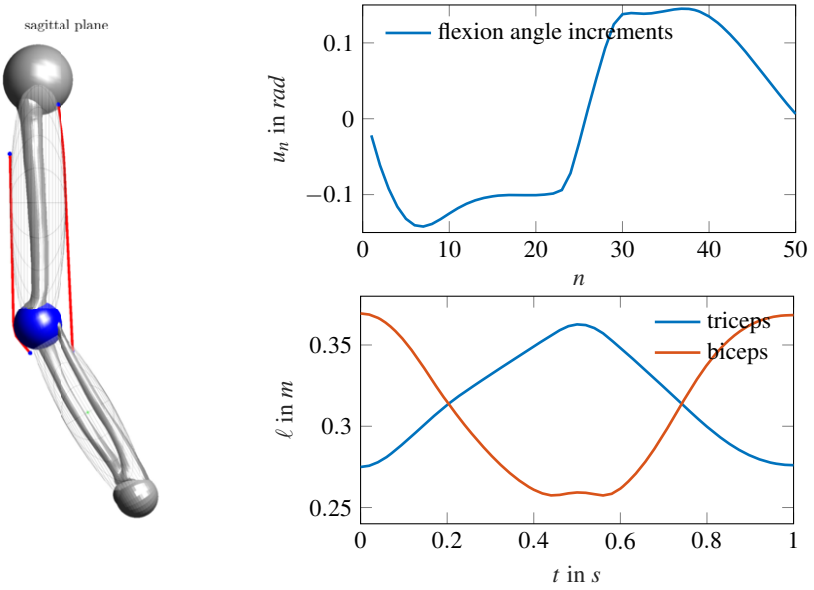

Figure 1. Configuration, joint angle change and muscle length evolution of the two muscle arm model[2] for a rest-to-rest manoeuvre are shown. The muscle path is plotted in red and the revolute joint is marked in blue.

realized by Hill-type muscle forces, similar to [2]. We solve a rest-to-rest manoeuvre from a given initial state to a predefined final state. The gradient, Jacobian and Hessian information for the optimization problem in IPOPT is carried out by CasADi. The incremental joint angle change and the muscle length evolution are shown in Figure 1 on the right side. The results show that the integrator and the optimal control simulation with the automatic differentiation work as expected. Due to the structure preserving properties of the integrator, the simulation works robustly even with coarse discretizations.

3.2. Upper Body Kinematic Model

The second multibody system comprises a kinematic chain with $b = 9$ rigid bodies, 8 joints and 12 degrees of freedom. Figure 2 shows the numbering of the rigid body and the joint list. The torso ($\beta = 1$) is fixed in space and is connected to the right humerus ($\beta = 2$) and left humerus ($\beta = 6$) via spherical joints, with $\boldsymbol{\theta}_n^{\{2,6\}} \in \mathbb{R}^3$. The ulna of the right arm ($\beta = 3$) and left arm ($\beta = 7$) is connected to the respective humerus via revolute joints, where $u_n^{\{3,7\}} \in \mathbb{R}$. Between the right ($\beta = 4$) and left ($\beta = 8$) radius and ulna, a revolute joint with $u_n^{\{4,8\}} \in \mathbb{R}$ is defined respectively. The right hand ($\beta = 5$) and the left hand ($\beta = 9$) are connected via revolute joints with $u_n^{\{5,9\}} \in \mathbb{R}$ to the respective radius. The rotation of a revolute joint $\boldsymbol{\theta}_n = u_n \boldsymbol{n}$ follows with the definition of the joint axis $\boldsymbol{n}$ and the joint location of the α-th joint with respect to the β-th body is given as $\boldsymbol{\rho}_\alpha^\beta$. With respect to Eq. (7) the mapping for the fixed torso reads

$$\begin{bmatrix} \boldsymbol{x}^1_{s_{n+1}} \\ \boldsymbol{d}^1_{I_{n+1}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}^1_{s_n} \\ \boldsymbol{d}^1_{I_n} \end{bmatrix} \tag{10}$$

for $I = 1,2,3$. The mapping for the right arm with bodies $\beta = \{2,3,4,5\}$ reads

$$\begin{bmatrix} \boldsymbol{x}^\beta_{s_{n+1}} \\ \boldsymbol{d}^\beta_{I_{n+1}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}^{\beta-1}_{s_{n+1}} + \operatorname{cay}(\widehat{\boldsymbol{\theta}^{\beta-1}_{n+1}}) \cdot \left(\boldsymbol{\rho}^{\beta-1}_\beta - \operatorname{cay}(\widehat{\boldsymbol{\theta}^\beta_{n+1}}) \cdot \boldsymbol{\rho}^\beta_\beta \right) \\ \operatorname{cay}(\widehat{\boldsymbol{\theta}^\beta_{n+1}}) \cdot \boldsymbol{d}^\beta_{I_n} \end{bmatrix} \tag{11}$$

where $I = 1,2,3$. The mapping for the left arm with bodies $\beta = \{6,7,8,9\}$ follows in the same way. The definition of the kinematic tree – with all rigid bodies and joints – and with that the reparametrization completes the model setup. No additional assembly of a null space matrix is necessary. With that, the extension of the DMOCC method to kinematic trees with a relatively large number of bodies is now much easier than through manual implementation. The next step is to automate the derivation of muscle path equations in a similar way.

4. Discussion

This work focuses on the definition and implementation of the discrete Euler-Lagrange equations for open kinematic chains with automatic differentiation, which are used to de-

[2]For the 3d bone model see https://www.thingiverse.com/thing:1543880.

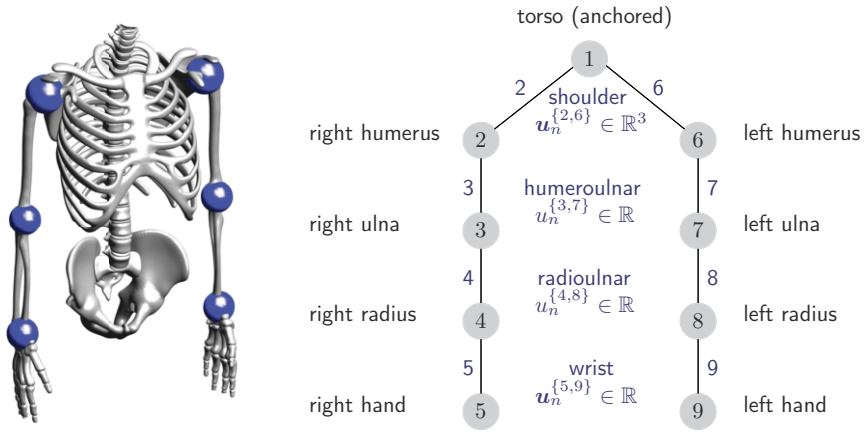

Figure 2. The configuration and the kinematic tree for the upper body model[2] is shown. The model comprises 9 rigid bodies with 12 degrees of freedom.

fine optimal control problems with DMOCC. All necessary derivatives were carried out with the help of the nonlinear optimization and algorithmic differentiation tool CasADi. This not only simplifies the solution to the optimization problem, but also the model creation benefits from that. In particular, we use a user defined reparametrization to calculate the discrete null space matrix for which no separate implementation is required. This simplifies and accelerates the creation and solution of the optimization problems transcribed by DMOCC.

By using CasADi to calculate the discrete null space matrix, it is possible to automate and simplify one of the most challenging tasks in our model assembly. Usually, the null space matrix $\boldsymbol{P}_d(\boldsymbol{q}_n) \in \mathbb{R}^{12b \times DoF}$ has to be assembled by traversing piecewise through the kinematic chain or has to be implemented by hand. Furthermore, the discrete Euler-Lagrange equations in Eqs. (2) and (3) are based on a nodal reparametrization $\boldsymbol{F}_d \in \mathbb{R}^{12b}$, which defines a suitable null space matrix by calculating its Jacobian. Thus, an additional implementation of the $12b \times DoF$ dimensional null space matrix is no longer necessary. Since the reparametrization remains definable by the user, our formulation also remains very flexible.

However, the presented method is limited to open kinematic chains and difficulties may arrive for practical application with inherent closed chains in the human skeleton, e.g. the shoulder girdle, forearm, grasping. However, for closed kinematic chains, one can formulate the open loop system a null space matrix and formulate an additional closure constraint. This procedure will still be much more efficient than formulating the complete system without a null space reduction. Our next steps include musculoskeletal optimal control simulations with a relative large number of rigid bodies (for example the upper body), muscles and general smooth wrapping surfaces that perform certain movement tasks. Therefore, we will automate the derivation of muscle path equations and contact formulation in a similar way.

Acknowledgements

The authors acknowledge the financial support by the German Federal Ministry of Educaiton and Research of Germany in the framework of DYMARA (project number 05M16WEB).

References

[1] Delp SL, Anderson FC, Arnold AS, Loan P, Habib A, John CT, Guendelman E and Thelen DG. OpenSim: Open-Source Software to Create and Analyze Dynamic Simulations of Movement. IEEE Transactions on Biomedical Engineering, November 2007, 54(11):1940–1950.

[2] Maas R, Leyendecker S. Biomechanical optimal control of human arm motion. Proceedings of the Institution of Mechanical Engineers, Part K: Journal of Multi-body Dynamics, December 2013, 227(4):375–389.

[3] Scholz A, Sherman M, Stavness I, Delp S, Kecskeméthy A. A fast multi-obstacle muscle wrapping method using natural geodesic variations. Multibody System Dynamics, 2016, 36(2):195–219.

[4] Marsden JE, West W. Discrete mechanics and variational integrators. Acta Numerica, 2001, 10:357–514.

[5] Betsch P, Leyendecker S. The discrete null space method for the energy consistent integration of constrained mechanical systems. Part II: Multibody dynamics. International journal for numerical methods in engineering, 2006, 67(4):499–552.

[6] Leyendecker S, Marsden JE, Ortiz M. Variational integrators for constrained dynamical systems. ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik: Applied Mathematics and Mechanics, 2008,88(9):677–708.

[7] Penner J, Leyendecker S. Multi-Obstacle Muscle Wrapping Based on a Discrete Variational Principle. In: Faragó I, Izsák F, Simon P, editos. Progress in Industrial Mathematics at ECMI 2018, volume 30, Cham, 2018: Springer International Publishing, p. 223–229.

[8] Penner J, Leyendecker S. A Hill Muscle Actuated Arm Model with Dynamic Muscle Paths. In: Kecskeméthy A, Geu Flores F, editos. Multibody Dynamics 2019, volume 53, Cham, 2019: Springer International Publishing, p. 52–59.

[9] Johnson ER, Murphey TD. Discrete and continuous mechanics for tree representations of mechanical systems. In 2008 IEEE International Conference on Robotics and Automation, Pasadena, CA, USA, May 2008, p. 1106–1111.

[10] Johnson ER, Murphey TD. Scalable Variational Integrators for Constrained Mechanical Systems in Generalized Coordinates. IEEE Transactions on Robotics, December 2009, 25(6):1249–1261.

[11] Björkenstam S, Leyendecker S, Linn J, Carlson JS, Lennartson B. Inverse Dynamics for Discrete Geometric Mechanics of Multibody Systems With Application to Direct Optimal Control. Journal of Computational and Nonlinear Dynamics, October 2018, 13(10):101001.

[12] Giftthaler M, Neunert M, Stäuble M, Frigerio M, Semini C, Buchli J. Automatic differentiation of rigid body dynamics for optimal control and estimation. Advanced Robotics, November 2017, 31(22):1225–1237.

[13] Leyendecker S, Ober-Blöbaum S, Marsden JE, Ortiz M. Discrete mechanics and optimal control for constrained systems. Optimal Control Applications and Methods, 2010, 31(6):505–528.

[14] Andersson J, Gillis J, Horn G, Rawlings JB, Diehl M. CasADi: a software framework for nonlinear optimization and optimal control. Mathematical Programming Computation, March 2019, 11(1):1–36.

[15] Wächter A, Biegler LT. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical Programming, March 2006, 106(1):25–57.

[16] Ugray Z, Lasdon L, Plummer J, Glover F, Kelly J, Martí R. Scatter Search and Local NLP Solvers: A Multistart Framework for Global Optimization. INFORMS Journal on Computing, August 2007, 19(3):328–340.