Simulation of Constrained Musculoskeletal Systems in Task Space

Dimitar Stanev, and Konstantinos Moustakas, *Senior Member, IEEE*

***Abstract*—*Objective:* This work proposes an operational task space formalization of constrained musculoskeletal systems, motivated by its promising results in the field of robotics. *Methods:* The change of representation requires different algorithms for solving the inverse and forward dynamics simulation in the task space domain. We propose an extension to the Direct Marker Control and an adaptation of the Computed Muscle Control algorithms for solving the inverse kinematics and muscle redundancy problems respectively. *Results:* Experimental evaluation demonstrates that this framework is not only successful in dealing with the inverse dynamics problem, but also provides an intuitive way of studying and designing simulations, facilitating assessment prior to any experimental data collection. *Significance:* The incorporation of constraints in the derivation unveils an important extension of this framework towards addressing systems that use absolute coordinates and topologies that contain closed kinematic chains. Task space projection reveals a more intuitive encoding of the motion planning problem, allows for better correspondence between observed and estimated variables, provides the means to effectively study the role of kinematic redundancy and, most importantly, offers an abstract point of view and control, which can be advantageous towards further integration with high level models of the precommand level. *Conclusion:* Task-based approaches could be adopted in the design of simulation related to the study of constrained musculoskeletal systems.**

***Index Terms*—task space, musculoskeletal system, inverse dynamics, forward dynamics, constrained mechanics.**

I. Introduction

THE human coordination problem combines many levels of hierarchical organization that are performed in our body [1], [2]. One way to model the complexity of controlling the musculoskeletal system [3] is to partition the problem into layers of abstraction and to encode the physiological factors that describe their functionality, providing the necessary details for the particular problem. Frequently, we find it difficult to express an abstract idea that encapsulates a problem and relate it to the underlying mathematical model. This has partly contributed to the fact that most of the algorithms are expressed in joint space, which may not always be the optimal representation.

As a matter of fact, many applications require a change of representation to solve a particular problem. Relevant examples include: the study of the mechanical capabilities of the human limbs [4], where fundamental questions on the interaction of the nervous system with the physical world are addressed; the role of muscle synergies, their relation to the execution of a task [5] and the influence they have in shaping the end point stiffness [6]; predictive simulation of closed-chain systems during execution of a crank-rotation task [7]. In this context, researchers are interested in relating the meaningful, observable quantities (e.g. position, orientation, force, end point stiffness, etc.) exerted by a segment (typically end-effector) to the muscle recruitment distribution. To account for this, we seek to establish the equations that dictate the task characteristics and muscle excitation patterns.

The main goal of this work is to provide a transparent, bidirectional link between the task and the corresponding muscle coordination patterns. The concept of a task has an abstract meaning and can be interpreted as a high level command that encapsulates the characteristics of a movement similar to the planning occurring in the brain [8]. The present study focuses mainly on motion primitive tasks (e.g. position and orientation), without necessarily restricting applications to those alone (e.g. force or impedance control [9] could also be addressed). The operational task space formalization [10] provides the mathematical tools to map the Equations of Motion (EoM) of the underlying system to the space of the task and control the movement in this domain.

The proposed algorithms and controllers are designed so as to handle constrained multibody systems. This is of significant importance since many biological models (e.g. anatomical joints [11]) or experimental setups (e.g. crank-rotation tasks [7]) are modeled through constraints. Consequently, the presented framework can address models that use absolute (Cartesian) coordinates [12] instead of generalized coordinates (Subsection S-D[1]) or systems that contain closed kinematic chains [13] (Subsection S-E). The effect of different joint topologies (e.g. closed kinematic chains) and the influence of constraint modeling on the required muscle forces remains a subject open to study. Results show that constraint modeling can alter the required generalized motion forces and consequently affect the muscle forces (Subsection III-A).

Although task space formalization has been proposed for the study of musculoskeletal systems [14], there is a lack of tools suitable for solving the inverse and forward dynamics problems [15]. Considering this, we present an implementation of a dynamically-based Inverse Kinematics (IK) algorithm (Task Space Dynamic Inverse Kinematics (TSDIK) Subsection II-G) for solving the IK problem. We show that this algorithm is capable of tracking the recorded movement (Subsection III-B),

Manuscript received July 07, 2017; accepted October, 15, 2017.

This work is partially funded by the EC FP7 project NoTremor: Virtual, Physiological and Computational Neuromuscular Models for the Predictive Treatment of Parkinsons Disease, Grant Agreement No. 610391.

D. Stanev and K. Moustakas are with the Department of Electrical and Computer Engineering, University of Patras, Greece, 26504 e-mail: (stanev, moustakas)@ece.upatras.gr

[1]The "S-" prefix denotes a reference to a supplementary material.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

while satisfying the imposed constraints due to its dynamic nature. Moreover, we present an algorithm (Task Space Computed Muscle Control (TSCMC) Subsection II-H) for performing muscle driven dynamics simulations that closely reproduce experimental measurements of kinematics and ground reaction forces, similar to the Computed Muscle Control (CMC) algorithm [16]. The latter is used to perform Forward Dynamics (FD) simulations by encoding the movement behavior from a set of abstract task goals, thus enabling model assessment prior to any experimental data collection. The performance of the two algorithms has been evaluated for a gait movement against the current available state of the art schemes.

We employed and extended the open source tool OpenSim [17] for modeling and simulation of the human movement. OpenSim represents the open science movement, algorithm sharing and reproducibility of scientific results [18]. Currently, OpenSim does not provide an integrated support for handling operational task space formulation and projection of the EoM (Subsection S-A).

A. Related Work

Operational task space [10], [19] control has been successfully applied by the robotics community yielding novel control schemes, intuitive design, implementation and practical applications. Multibody theory is the basis for studying the kinematics and dynamics of the skeletal system. The underlying EoM have been extended, so as to describe the human muscle complexity [20], [21]. Consequently, this can be used to evaluate the cause of an orchestrated muscle command to the observable movement. Simulation of human movement has been successfully translated in clinical treatment of cerebral palsy, lower extremity amputees and osteoarthritis [22]. As well as, basic science related to the understanding of movement progression and control during dynamic tasks, contributing to the perception of important neurodegenerative diseases [17], [23], [24]. Pioneer work in the combination of the above mentioned fields has been presented in [14], stressing how these concepts can be applied to study the musculoskeletal complex in the task space. The results have meaningful biological interpretations and are not a product of systems engineering.

In clinical practice, there are two main approaches for computing the internal state of the musculoskeletal composite from the available measurements [15]. During Inverse Dynamics (ID), the analysis starts from the effect (resultant movement) and propagates all the way up in the hierarchy to the cause (muscle excitation signals) [25]. On the other hand, during FD a synchronous command triggers the muscles, whereas the resulting movement is estimated and observed. Different combinations of these schemes can be found in the literature [26], [27], building upon the problem's complexity, the available measurements and the assessment requirements. Commonly observable variables are the movement of the segments recorded by a Motion Capture (MoCap) system, the external forces that act on the system (e.g. ground reaction during walking) [28] and the Electromyography (EMG) recordings of muscle activity [29], [30]. These measurements are used to estimate the internal state of the model (joint space generalized forces, muscle forces, etc.) and further restrict the possible solutions of the redundant system, both in terms of kinematic and dynamic redundancy.

Many biological models require adequate constraint modeling, so that the underlying model approximates the realistic movement behavior accurately [11]. The knee and shoulder complex are representative examples of such models [13], [31], [32]. The incorporation of constraints introduces many technical and scientific challenges when solving the forward and inverse dynamics problem [33]–[35]. Some of them are finding a solution existence of the system configuration under constraints and difficulties in solving the Ordinary Differential Equations (ODE) [36]. As an example, numerical drifts can still result in constraint violation [36], [37] and the introduction of constraints can result in Differential Algebraic Equations (DAE) of index-3 that are hard to solve [36]. Although constraints are an inseparable part of the model, their influence on the muscle recruitment problem is not thoroughly studied, even if any underlying simulation scheme should be able to handle these types of systems.

The proposed framework fits well into the wide spectrum of the current biomedical literature extending the task-based approach towards applications in musculoskeletal simulation. The developed algorithms solve the motion planning problem intuitively with direct translation in clinical practice. Extension towards constrained multibody systems facilitates the modeling of the broader type of problems found in many biological systems.

II. Methods

A. Preliminaries - Notation

Uppercase letters will be used to denote matrix quantities with specific dimensions (e.g. $C \in \Re^{n\times n}$). Vector quantities will be expressed by lowercase letters (e.g. $c \in \Re^m$).

Linear projection operators are used extensively, deeming the introduction of relevant notation necessary. For a given linear transformation

$$Ax = b,\ x \in \Re^n, b \in \Re^m$$

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} = \begin{bmatrix} | & & | \\ c_1 & \cdots & c_n \\ | & & | \end{bmatrix} = \begin{bmatrix} - & r_1^T & - \\ & \vdots & \\ - & r_m^T & - \end{bmatrix} \tag{1}$$

the matrix A defines a mapping $A : \Re^n \to \Re^m$. The column space (image or range) of A is a space spanned by its n m-D column vectors

$$\mathcal{C}(A) = span(c_1, \cdots, c_n) \subseteq \Re^m \tag{2}$$

which is a r-D ($r \leq n$ independent columns) subspace of $\Re^m$ composed of all possible linear combinations of its n column vectors. The row space of A is a space spanned by its m n-D row vectors

$$\mathcal{R}(A) = span(r_1, \cdots, r_m) \subseteq \Re^n \tag{3}$$

which is a r-D subspace of $\Re^n$ composed of all possible linear combinations of its m row vectors. The left null space of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

A ($\mathcal{N}(A)$), is the set of all x that satisfy the homogeneous equation

$$\mathcal{N}(A) = \{x \in \Re^n : Ax = 0\} \subseteq \Re^n \tag{4}$$

and similarly the right null space of A ($\mathcal{N}(A^T)$) is the set of all b that satisfy the following relation

$$\mathcal{N}(A^T) = \{b \in \Re^m : A^T b = 0\} \subseteq \Re^m \tag{5}$$

The following properties between the various subspaces hold for an operator A ($\perp$ stands for orthogonal complement and the symbol $\oplus$ denotes the direct sum)

$$\begin{gathered}\mathcal{R}(A) = \mathcal{C}(A^T),\ \mathcal{C}(A) = \mathcal{R}(A^T) \\ \mathcal{R}(A) \cap \mathcal{N}(A) = \emptyset,\ \mathcal{R}(A) \perp \mathcal{N}(A),\ \mathcal{R}(A) \oplus \mathcal{N}(A) = \Re^n \\ \mathcal{C}(A) \cap \mathcal{N}(A^T) = \emptyset,\ \mathcal{C}(A) \perp \mathcal{N}(A^T),\ \mathcal{C}(A) \oplus \mathcal{N}(A^T) = \Re^m\end{gathered} \tag{6}$$

The general solution of Eq. (1) in the underdetermined case ($n > m$), since this is how redundancy is presented, is of the form

$$x = x_{\parallel} + x_{\perp} \tag{7}$$

where $x_{\parallel} \in \mathcal{R}(A)$ is a particular solution and $x_{\perp} \in \mathcal{N}(A)$ is an arbitrary vector that lies in the direction of the null space of A ($Ax_{\perp} = 0$).

A projection operator P is a linear transformation from a vector space to itself ($P^2 = P$) and the operator is termed orthogonal if $P = P^T$ [38]. Note that the term "orthogonal projection" is a projection for which the range and the null space are orthogonal subspaces. There are two projection matrices associated with each vector space defined in Eq. (1), that can divide a vector into two perpendicular components. Namely the projection of x onto the column space of $A^T, \mathcal{C}(A^T)$, the projection of x onto the null space of $A, \mathcal{N}(A)$, the projection of b onto the column space of $A, \mathcal{C}(A)$ and the projection of b onto the null space of $A^T, \mathcal{N}(A^T)$ such that they span the entire space.

B. Equations of Motion of the Plant

In the derivation of the EoM, we will address systems with holonomic constraints as we can't ignore their importance in modeling a complex system. The fundamental equations of differential variational principles for a constrained mechanical system leads to the following DAE of index-3 [36]

$$\begin{aligned}M\ddot{q} + f &= \tau - \tau_c \\ f &= g + h + f_o\end{aligned} \tag{8}$$

$$\phi(q) = 0 \tag{9}$$

where $M \in \Re^{n \times n}$ is the symmetric, positive definite joint space inertia mass matrix, n are the number of Degrees of Freedom (DoFs) of the model and $q, \dot{q}, \ddot{q} \in \Re^n$ are the joint space generalized coordinates and their derivatives with respect to time. The term $f \in \Re^n$ is the sum of all joint space forces, $g \in \Re^n$ is the gravity, $h \in \Re^n$ the Coriolis and centrifugal and $f_o \in \Re^n$ other generalized forces[2]. Term $\tau \in \Re^n$ is a vector of applied generalized forces that actuate the model and $\tau_c \in \Re^n$ represents the generalized forces induced by constraints. Eq. (9) corresponds to a set of c constraint algebraic equations which can be differentiated twice with respect to time (the dot notation depicts a derivative with respect to time)

$$J_c \dot{q} = 0, J_c = \frac{\partial \phi}{\partial q} \tag{10}$$

$$J_c \ddot{q} = -\dot{J}_c \dot{q} = b \tag{11}$$

where the term $J_c \in \Re^{c \times n}$ defines the constraint Jacobian.

At this point it is worth considering the effect of equations (9), (10) and (11) on the permissible configuration solutions (Eq. (8)). The algebraic constraint equations (Eq. (9)) imply that any admissible configuration $q \in \Re^n$ should lie on the constraint manifold [13], [33]. The first derivative of the constraints (Eq. (10)) suggests that $\dot{q} \in \mathcal{N}(J_c)$, following the definition of null space (Eq. (4)). As a consequence of the D'Alembert's principle, constraint forces do no work under any virtual displacement that satisfy them ($\tau_c \delta q = 0, \forall \delta q \in \mathcal{N}(J_c)$ which is equivalent to $\tau_c \perp \delta q, \forall \delta q \in \mathcal{N}(J_c)$). In turn, this implies that $\tau_c \in N(J_c)^{\perp} = \mathcal{C}(J_c^T)$. Thus, the generalized constraint forces can be represented as a linear combination of columns of J_c^T

$$\tau_c = J_c^T \lambda \tag{12}$$

where $\lambda \in \Re^c$ stands for the vector of Lagrange multipliers. While $\dot{q}_{\parallel} = 0$, where $\dot{q}_{\parallel} = \{\dot{q} : \dot{q} \in \mathcal{N}(J_c)^{\perp}\}$, the same does not always hold for the acceleration ($\ddot{q}_{\parallel} \neq 0$) by inspecting the second derivative of the constraints (Eq. (11)).

Eq. (8) can be solved in a FD manner for the unknowns $\ddot{q}, \lambda$, through the complement of equations (9), (10) and (11). The constraint derivatives can reduce the DAE to index-1, making the system solvable by an ODE solver. However, numerical integration inevitably leads to drifts that eventually result in constraint violation ($\phi(q(t)) \neq 0$). Coordinate projection [36] or Baumgarte's stabilization [37] can be introduced to ensure exponential elimination of the constraint error, given the appropriate initial conditions.

C. Inverse Dynamics Controller

Before presenting the Inverse Task Space Controller (ITSC) we will first elaborate on alternative implementations of the constrained ID model that can significantly impact the derivation of the required generalized forces. A consequence of incorporating constrains is that if the system is over-constrained, different sets of applied forces can generate the same movement behavior [34]. In the following, we will present two types of controllers, that project the underlying EoM into the constraint manifold by two different projection operators. The derivation begins with the following model

$$M\ddot{q} + f + J_c^T \lambda = \tau \tag{13}$$

$$J_c \ddot{q} = b \tag{14}$$

[2]Most of the quantities in the equations are a function of the generalized coordinates q and their derivatives (e.g. $M(q), f(q, \dot{q})$). This dependency will be omitted for simplicity.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

Both controllers aim to decouple the constraint and applied forces. The first approach [13] eliminates the constraints by solving equations (13) and (14) for λ

$$\begin{aligned} \underbrace{J_c M^{-1} M \ddot{q}}_{b} + J_c M^{-1} f + \underbrace{J_c M^{-1} J_c^T}_{\Lambda_c^{-1}} \lambda = J_c M^{-1} \tau \\ \lambda = \underbrace{\Lambda_c J_c M^{-1}}_{\bar{J}_c^T} (\tau - f) - \Lambda_c b \end{aligned} \tag{15}$$

where $\Lambda_c \in \Re^{c \times c}$ is the constraint compliant inertia mass matrix and $\bar{J}_c \in \Re^{n \times c}$ is the generalized inertia weighted inverse of the constraint Jacobian (J_c). We can substitute the solution obtained from Eq. (15) into Eq. (13) (Model-A):

$$\begin{aligned} M\ddot{q} + f_\perp + b_c = \tau_\perp \\ f_\perp = N_c^T f, \tau_\perp = N_c^T \tau \\ N_c^T = I - J_c^T \bar{J}_c^T \\ b_c = -J_c^T \Lambda_c b \end{aligned} \tag{16}$$

where $N_c \in \Re^{n \times n}$ is the constraint null space matrix ($\mathcal{C}(N_c) = \mathcal{N}(J_c)$) and $b_c \in \Re^n$ is a force vector induced by the constraint acceleration bias term (Eq. (14)). Here, the symbol $\perp$ denotes that the projected quantities lie in the null space of the constraints. This signifies that only the null space component of the generalized forces contributes to the motion of a constrained mechanical system, while the residual forces ($(I - N_c^T)(\tau - f)$) are compensated by the constraints. Moreover, N_c is a projection operator ($N_c^2 = N_c$), which is not necessarily orthogonal ($N_c \neq N_c^T$) and $J_c N_c = 0$ holds by virtue of the relation between the null and range space (Eq. (6)).

In contrast to Model-A, Aghili [33] proposed a more general derivation of the constrained ID model. The EoM can be projected into the null space of the constraints, eliminating the constraint forces through an orthogonal projection operator $N_c^{'}$ ($N_c^{'2} = N_c^{'}, N_c^{'} = N_c^{'T}$), where $J_c N_c^{'} = 0$ and $N_c^{'} = I - J_c^{+} J_c$ ($+$ indicates the Moore-Penrose pseudoinverse). In comparison with the projection operator N_c, that depends on the inertia mass matrix, $N_c^{'}$ is easier to compute in practice [34], being purely kinematic dependent. Equations (13) and (14) can be expressed as follows

$$N_c^{'} M \ddot{q} + N_c^{'} f = N_c^{'} \tau \tag{17}$$

$$\ddot{q}_\parallel = (I - N_c^{'}) \ddot{q} = J_c^{+} b \tag{18}$$

Typically $N_c^{'}$ is rank deficient, thus $N_c^{'} M$ is not invertible. However, since the system is constrained and equations (17) and (18) lie in mutually orthogonal spaces (they cannot cancel each other out), they can be added (Model-B):

$$\begin{aligned} M^{'} \ddot{q} + f_\perp + b_c^{'} = \tau_\perp \\ M^{'} = M + N_c^{'} M - (N_c^{'} M)^T \\ f_\perp = N_c^{'} f, \tau_\perp = N_c^{'} \tau \\ b_c^{'} = -M J_c^{+} b \end{aligned} \tag{19}$$

where the choice of $M^{'}$ is not unique, as there are many ways that equations (17) and (18) can be combined. In the derivation of Eq. (19) they choose to premultiply Eq. (18) by M before adding to Eq. (17). It can be shown that $M^{'}$ is invertible as long as M is invertible [33]. It is evident that the choice of $b_c^{'}$ depends on the choice of $M^{'}$.

The constraint generalized forces (τ_c) can be derived by first projecting Eq. (13) with $(I - N_c^{'})$ and combine the result from Eq. (19)

$$\begin{aligned} (I - N_c^{'}) M \ddot{q} + (I - N_c^{'})(f + \tau_c) = (I - N_c^{'}) \tau \\ (I - N_c^{'}) M M^{'-1} (\tau_\perp - f_\perp - b_c^{'}) + f_\parallel + \tau_c = \tau_\parallel \\ \tau_c = \tau_\parallel - f_\parallel - \mu(\tau_\perp - f_\perp - b_c^{'}) \end{aligned} \tag{20}$$

where $\mu = (I - N_c^{'}) M M^{'-1}$ and $\tau_\parallel, f_\parallel$ represent the contribution of these forces in the constraint subspace. Eq. (20) implies that the constraint generalized forces (τ_c) can be obtained uniquely, whereas the derivation of the Lagrange multipliers (λ) is not unique, since J_c can be rank deficient.

D. Inverse Task Space Controller for Single Task

In this section we will describe the process of projecting the EoM derived previously into the space of the task. In this derivation we will not consider any prior choice of M, b_c, N_c.

Each task is associated with a task Jacobian matrix $J_t \in \Re^{d \times n}$, where d is the dimension of the task (e.g. three for position task, three for orientation task and six for spatial task, which is the combination of the first two)[3]. For a fixed joint space configuration, J_t is a linear mapping from joint velocities $\dot{q}$ to task velocities $\dot{x}_t$

$$\dot{x}_t = J_t \dot{q} \tag{21}$$

$$\ddot{x}_t = \dot{J}_t \dot{q} + J_t \ddot{q} \tag{22}$$

In turn, the transpose of a task Jacobian maps from task space forces $f_t \in \Re^d$ to joint space generalized forces $\tau \in \Re^n$

$$\tau = J_t^T f_t \tag{23}$$

For a given configuration, there is an infinite number of elementary displacements that could take place without altering the configuration of the effector [10], because $d < n$ in general. Those displacements correspond to motion in the null space associated with the generalized inverse of the Jacobian matrix. With the addition of the null space forces, the relationship between task forces and manipulator joint space generalized forces takes the following general form

$$\tau = J_t^T f_t + N_t^T \tau_0, \; N_t^T = (I - J_t^T \bar{J}_t^T) \tag{24}$$

where $N_t \in \Re^{n \times n}$ is the null space of J_t ($J_t N_{J_t} = 0$), $\tau_0 \in \Re^n$ is a vector of arbitrary selected generalized forces, which will be projected in the null space direction of J_t and $\bar{J}_t \in \Re^{n \times d}$ is the generalized inverse of J_t.

Khatib [10] showed that the generalized inverse for a single task and an unconstrained system exists and is uniquely defined as follows

$$\bar{J}_t^T = \Lambda_t J_t M^{-1} \tag{25}$$

[3]Note that the dimension of the task can be $1 \leq d \leq 6$, since in general we can choose which DoFs to utilize. A task that does not have DoFs ($d = 0$) is a constraint.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

$$\Lambda_t = (J_t M^{-1} J_t^T)^{-1} \tag{26}$$

where $\Lambda_t \in \Re^{d\times d}$ is the task compliant inertia mass matrix. The dynamically consistent generalized inverse ($\bar{J}_t$) is the only generalized inverse that results in zero task acceleration for any τ_0. In the presence of a single task the control law has the form of Eq. (24). We can apply this control law to the constrained equations of the previous section and solve for the task forces

$$\begin{aligned} \underbrace{J_t M^{-1} M\ddot{q}}_{\ddot{x}_t - \dot{J}_t\dot{q}} + J_t M^{-1}(f_\perp + b_c) &= \underbrace{J_t M^{-1} N_c^T J_t^T}_{\Lambda_{t|c}^{-1}} f_t \\ \Lambda_{t|c}(\ddot{x}_t + b_t) + \bar{J}_{t|c}^T(f_\perp + b_c) &= f_t \\ J_t M^{-1} N_c^T (I - J_t^T \bar{J}_t^T)\tau_0 &\stackrel{!}{=} 0 \end{aligned} \tag{27}$$

where $\Lambda_{t|c} \in \Re^{d\times d}$ is the constrained task compliance inertia mass matrix, $b_t = -\dot{J}\dot{q}$ is the task acceleration bias term and $\bar{J}_{t|c}^T = \Lambda_{t|c} J_t M^{-1}$ is the constrained task Jacobian generalized inverse. The symbol $t|c$ denotes that the task t quantity is "filtered out" (prioritized) by the null space of the constraints c. Note that we required that the null space forces τ_0 should not interfere with the task acceleration. To achieve this for any $\tau_0 \neq 0$ we must solve equation $J_t M^{-1} N_c^T (I - J_t^T \bar{J}_t^T) = 0$ for $\bar{J}_t^T$

$$\bar{J}_t^T = \bar{J}'^T_{t|c} = \Lambda_{t|c} J_t M^{-1} N_c^T \neq \bar{J}_{t|c}^T \tag{28}$$

where, in turn the total residual null space of the task subject to the constraints is given by

$$N_{t|c}^T = N_c^T (I - J_t^T \bar{J}'^T_{t|c}) = (I - J_{t|c}^T \bar{J}_{t|c}^T) N_c^T \tag{29}$$

where $J_{t|c}^T = N_c^T J_t^T$. Finally, the computed generalized forces for a given desired command $\ddot{x}_t$ are

$$\tau_\perp = J_{t|c}^T f_t + N_{t|c}^T \tau_0 \tag{30}$$

E. Inverse Task Space Controller for Multiple Tasks

An important aspect of working in the task space is that in the presence of multiple tasks [39], we can further assign a priority level for each task. Consequently, lower level tasks will not produce acceleration that will interfere with the performance of higher priority tasks. This is achieved by projecting the individual task into the null space of the higher priority tasks through the null space matrix. This is very important for two reasons: 1) helps to assign an importance factor in the execution of tasks and 2) helps to further partition and decompose the control into individual tasks that do not interfere with higher priority ones. The following control scheme is adopted

$$\tau = \underbrace{\sum_{i=1}^{g} J_{i|i-1*}^T f_i}_{\text{task forces}} + \underbrace{N_{g*}^T \tau_0}_{\text{null space forces}} \tag{31}$$

There are g tasks in total and each task force (f_i) is projected onto the aggregate null space ($J_{i|i-1*} = J_i N_{i-1*}$) of the higher priority tasks with respect to i ($i*$ denotes an aggregation). For the purpose of controlling the musculoskeletal system we partition the applied generalized forces (τ), into those that are responsible for driving the task goals and those that operate in the aggregate null space of all tasks (N_{g*}^T). This notation is very similar to the posture control separation proposed in [39], but we can choose to utilize the kinematic redundancy in a different manner. More specifically, the term τ_0 can be appropriately selected to compensate for any residual force in the null space (e.g. $\tau_0 = f_\perp + b_c$). Alternatively, this redundancy can be exploited to solve the case of underactuated systems [33], [35]. Let's apply Eq. (31) to the ID model and solve for any arbitrary task force $f_k, 1 \leq k \leq g$

$$\begin{aligned} J_k M^{-1} M\ddot{q} + J_k M^{-1}(f_\perp + b_c) &= \underbrace{J_k M^{-1} J_{k|k-1*}^T}_{\Lambda_{k|k-1*}^{-1}} f_k \\ &+ J_k M^{-1} \sum_{i\neq k}^{g} J_{i|i-1*}^T f_i + \cancelto{0}{J_k M^{-1} N_{g*}^T \tau_0} \\ \Lambda_{k|k-1*}(\ddot{x}_k + b_k) + \Lambda_{k|k-1*} J_k M^{-1}(f_\perp + b_c) &= f_k \\ &+ \Lambda_{k|k-1*} J_k M^{-1} \sum_{i=1}^{k-1} J_{i|i-1*}^T f_i \\ f_k = \Lambda_{k|k-1*}(\ddot{x}_k + b_k) &+ \bar{J}_{k|k-1*}^T (f_\perp + b_c - \tau_{k-1*}) \end{aligned} \tag{32}$$

where τ_{k-1*} is the contribution of the higher priority tasks. Note that the summation term, which represents the force contribution of each task, stops at index $k-1$ (not g) because all lower priority tasks ($i > k$) will yield zero interference, complying with our requirement. The same applies to the null space term $J_k M^{-1} N_{g*}^T \tau_0 = 0$. It can be shown that the null space of task k, subject to the higher priority tasks, is defined in a similar manner to the single task case.

$$\begin{aligned} N_{k*}^T = N_{k-1*}^T (I - J_k^T \bar{J}'^T_{k|k-1*}) &= (I - J_{k|k-1*}^T \bar{J}_{k|k-1*}^T) N_{k-1*}^T \\ \bar{J}'^T_{k|k-1*} &= \Lambda_{k|k-1*} J_k M^{-1} N_{k-1*}^T \end{aligned} \tag{33}$$

F. Implementation

Previously (Subsection II-C) we decoupled the motion and constraint forces. For the rest of this study, we will focus on motion-based control schemes.

By working in task space, the input to the controller is a set of desired task goals. For a given set of goals, we can compute the task forces and map them to joint space forces through Eq. (31). The iteration starts with the higher priority task and propagates through the task priority graph $\mathcal{G}$, that is a priority ordered set (Algorithm 1) [13].

G. Task Space Dynamic Inverse Kinematics

There are a couple of ways to perform IK given a set of experimental recordings. The first is to use an optimization-based approach that minimizes an objective function. On the other hand, dynamics (Eq. (23)) can be utilized by computing a set of joint space forces that drive the model, which is subsequently integrated numerically. Optimization-based solutions can achieve better tracking in terms of the Root Mean Square (RMS) marker error. In contrast, the dynamic solutions can take advantage of the structure of the model, e.g. by addressing different types of constraints that are commonly handled at a dynamic level.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

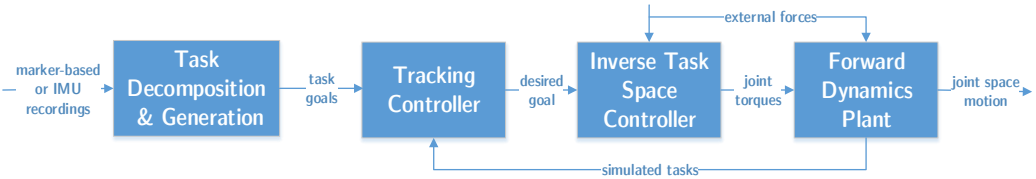


Fig. 1: Architecture diagram of the TSDIK method. The input to the system is a set of marker trajectories or other types of measurements (e.g. Inertial Measurement Unit (IMU)). Markers are grouped based on their attachment location on the body and a set of task goals is extracted from the recorded motion. A tracking scheme is used to track the task goals (e.g. Proportional Derivative (PD) controller) and the desired goals are transformed into generalized forces by the ITSC. Finally, these forces actuate the model in a FD manner. The product of this algorithm is both a set of task goals and the simulated motion that reproduces the observations.

Algorithm 1 Computes the driving torques for a set of goals.

Input: $\ddot{x}_t, \forall t \in \mathcal{G}$
Output: τ, τ_c
Choose M, b_c, N_c model
$N_{0*} = N_c$
$\tau_0 = 0$
for $i \in \mathcal{G}$ **do**
  Compute f_i from Eq. (32)
  $\tau_i = \tau_{i-1*} + J^T_{i|i-1*} f_i$
  Update N_{i*} from Eq. (33)
end for
$\tau = \tau_g + N^T_{g*}\tau_0$
Compute τ_c from equations (12) and (15) or (20)
return τ, τ_c

Recently, different types of MoCap systems have been employed to record the movement of the subject. In this derivation, it is possible to combine marker-based and IMU (Inertial Measurement Unit) based MoCap systems to record the spatial properties of body segments. The following implementation, inspired by the approach suggested by [14] (Direct Marker Control (DMC)), is enriched with some major improvements/modifications in order to accommodate for applications on more demanding movements (e.g. walking).

As presented in Fig. 1, the input is the trajectory of the markers and/or spatial motion of the body segments in the case of an IMU based system. The task decomposition and generation module is responsible for constructing the desired task goals based on one or more marker per body, or to prescribe the movement of the segment if IMU measurements are available. The desired tasks are used by the ITSC to compute the generalized forces that actuate the model. For a given actuation, the plant is numerically integrated and advanced in time. The resulting simulated tasks are compared with the measured ones and corrections are performed by the tracking controller.

In case of a marker based system, the marker trajectories are recorded and the corresponding virtual markers are attached to the body segments, each containing zero or more markers. We denote the position of the markers attached to a body b and measured in the ground frame G at time t as

$$\begin{gathered} {}^G p^b_{:}(t) = \{{}^G p^b_1(t), \ldots, {}^G p^b_{m_b}(t)\} \\ {}^G p^b_i(t) \in \Re^3 \end{gathered} \tag{34}$$

In order to generate a set of task goals, the markers are grouped per body segment and a tracking task is assigned. Some segments don't contain any marker, while others can have less than three markers. To define the 6D motion of a segment in space, at least three markers per body are required. Fortunately, since some segments are constrained with respect to their position in the body tree kinematic chain, their configuration can be determined even with less markers.

If a body contains a single marker, then a position type task is assigned ($x(t) = \{p(t)\} \in \Re^3$). If a segment can only rotate relative to its parent body, then an orientation type task is assigned ($x(t) = \{\theta(t)\} \in \Re^3$). When a segment is permitted to move freely (e.g. the floating base of the model) a spatial type task is assigned ($x(t) = \{\theta(t), p(t)\} \in \Re^6$). As shown in Fig. 2, which depicts two frames of gait, a 6D spatial task is assigned to the pelvis and a 3D orientation task is assigned to the femur and the tibia.

Regardless of the task type, the final goal is a desired acceleration that the task should achieve. In order to compute the desired goal a PD tracking controller is adopted

$$a(t) = a_d(t) + k_p(x_d(t) - x(t)) + k_d(v_d(t) - v(t)) \tag{35}$$

where $a(t)$ is the task goal acceleration, $x_d(t), v_d(t), a_d(t)$ are the desired position, velocity and acceleration that are derived from the experimental marker trajectories, $x(t), v(t)$ are the current position and velocity of the task and k_p, k_d are the tracking gains.

In order to track the goal effectively, the first (${}^G v^b_i(t)$) and the second (${}^G a^b_i(t)$) derivatives of the marker position have to be evaluated (e.g. by fitting smooth splines to the recorded trajectories).

Estimating the orientation of the segment and the higher order derivatives from the marker positions requires a different strategy. We seek to find a transition, transformation matrix that maps a set of marker positions on frame n to a set of positions on the next frame $(n + 1)$. The optimal rotation and translation between the corresponding 3D points can be obtained from the Kabsch algorithm [40] (Algorithm S-2).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

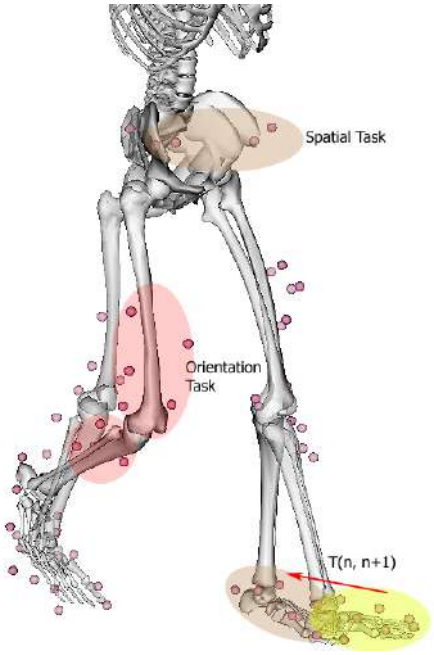


Fig. 2: This figure presents two frames of recorded gait. An ellipse is placed to denote the grouping of the markers, where the color defines the type of the task assigned to each group as an example. The transition, transformation ($T(n, n+1)$) between the two frames for the foot segment has been annotated.

By precomputing the transition, transformation matrix ${}^{G}T^{b}(n, n+1)$ and for a given initial pose ${}^{G}T^{b}(0, 1)$, the transformation of frame k with respect to the ground frame can be evaluated.

$$ {}^{G}T^{b}(k) = {}^{G}T^{b}(k-1, k) \cdot \ldots \cdot {}^{G}T^{b}(0, 1) \tag{36} $$

Unfortunately, the quantities of interest are the angular velocity ${}^{G}\omega^{b}(t)$ and angular acceleration ${}^{G}\alpha^{b}(t)$ of the frame. Quaternion representation is used, so that interpolation between consequent frames is more natural. For the interpolation a smooth curve should be always situated on the unit quaternion hypersphere $||q|| = 1, q \in \Re^4$ (here q is a quaternion). By constructing quaternion splines that are twice differentiable in $\mathcal{C}^2$ [41], the higher order derivatives of the orientation $q(t), \dot{q}(t), \ddot{q}(t)$ can be computed. Then the physical quantities of the rotational motion are deduced

$$ \omega = 2\dot{q}q^{*} \tag{37} $$

$$ \alpha = 2(\ddot{q}q^{*} - (\dot{q}q^{*})^{2}) \tag{38} $$

where $q^{*} = q_0 - q_1\hat{i} - q_2\hat{j} - q_3\hat{k}$ is the quaternion conjugate.

In Eq. (38) the power property of a quaternion is used to compute the term $(\dot{q}q^{*})^{2}$ (quaternions are closed over the product operator)

$$ \begin{aligned} q &= |q|e^{\hat{u}\theta} \\ q^{p} &= |q|^{p}(e^{\hat{u}\theta})^{p} = |q|^{p}(cos(p\theta) + \hat{u}sin(p\theta)) \end{aligned} \tag{39} $$

where $\hat{u}$ depicts a direction. After the task goals are precomputed, the ITSC (Subsection II-F) can be used to derive the necessary joint space.

H. Task Space Computed Muscle Control

As illustrated in Fig. 3 the algorithm can handle a set of desired task goals. These goals may originate from a high level controller that encapsulates the logic of producing a synchronous movement. Alternatively, the goals computed by TSDIK can be used as input. The output of the algorithm consists of the observable variables, such as factors related to the tasks (task Jacobian, task compliant inertia mass matrix, task forces, etc.), joint space forces, muscle forces, excitation patterns and the simulated movement. The flow begins with the task goals, which are compared against the actual (simulated) goals and a corrective action is issued to the ITSC. In turn, the ITSC computes the joint space forces, accounting for any prioritization scheme between the individual tasks. These forces are projected onto muscle excitation patterns through an optimization procedure (Subsection II-I). Finally, the plant is numerically integrated and advanced in time for the given command and initial states.

In contrast to the proposed algorithm, the CMC [16] accepts joint trajectories. The controller computes a set of muscle excitation patterns through Static Optimization (SO) [42] and drives the FD plant simulating a movement (joint space). In turn, the simulated trajectories are compared against the desired ones by a PD control law. In a very similar manner, instead of tracking trajectories, the TSCMC algorithm tracks task goals.

I. Muscle Effort Assessment

Typically, more than one muscles span a single DoF. Thus a unique mapping between muscle and joint space forces cannot be strictly defined. Formally, the relationship is given by

$$ \tau = R_m f_m \tag{40} $$

where $R_m \in \Re^{n \times m}$ is the moment arm fat matrix [43], [44], that maps from muscle space to joint space and $f_m \in \Re^{m}$ is a vector of muscle forces with m the number of muscles. In terms of solution existence, $m > n$, thus there is an infinite number of solutions of muscle forces that can produce the required generalized forces

$$ f_m = R_m^{+}\tau + (I - R_m^{+}R_m)f_o \tag{41} $$

where $f_0 \in \Re^{m}$ is an arbitrary force projected in the null space of R_m. In this study, we address the muscle redundancy (dynamic redundancy) problem through optimization.

The Hill type muscle model has been widely adopted by the biomechanics community [20], [21], [45] mainly for efficient simulation and reliable parameter estimation. The muscle force is governed by its activation and contraction dynamics

$$ \begin{aligned} \dot{a}_m &= g(u_m, a_m; \tau_a, \tau_d) \\ \dot{l}_{mt} &= h(q; \theta_0) \\ f_m &= d(a_m, l_{mt}, \dot{l}_{mt}; \theta_1) \end{aligned} \tag{42} $$

where a_m is the muscle activation, which is a function of the excitation u_m, τ_a, τ_d are the activation and deactivation time constants respectively. The musculotendon length l_{mt} and velocity $\dot{l}_{mt}$ are related to the generalized coordinates (q),

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

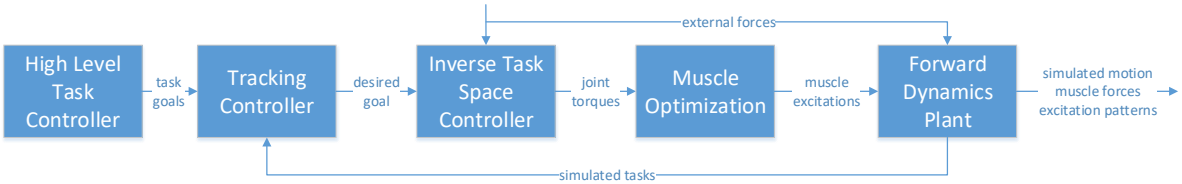


Fig. 3: Architecture diagram of the TSCMC. The input can be the task goals originating from the TSDIK or a set of goals from a different objective controller. The goals are tracked by an internal tracking scheme (e.g. PD controller), while the desired goals are used by ITSC to derive the required generalized forces. These forces are then mapped to muscle excitation patterns through an optimization procedure and the model is numerically integrated for the given muscle activation patterns.

while θ_0 stands for a multitude of parameters (e.g. pennation angle, maximum velocity and muscle routing). The muscle force f_m depends on these quantities, whereas θ_1 denotes another set of parameters such as maximum isometric force (the capacity of a muscle to produce force).

For a given set of desired goals that are produced by the ITSC, the muscles must equilibrate the required generalized forces (τ). Static optimization is a common technique for solving the muscle redundancy problem [24]. Unfortunately, models are imperfect and common modeling choices may introduce simulation instabilities. For a stable simulation the equality $\tau = R_m f_m(a)$ should hold true, but the muscles may not be actually able to produce the right amount of force, due to low strengths (maximum isometric force parameter). To avoid problems of this kind, a set of reserve actuators is introduced to compensate for this defect. Furthermore, their contribution is adjusted using a penalty factor, so as to minimize their use with respect to muscles

$$\begin{aligned} \min_{a(j),\beta(j)} \; & \frac{1}{m}\sum_{i}^{m} a_i^2(j) + \gamma \frac{1}{n}\sum_{i}^{n}\left(\frac{\beta_i(j)}{\beta_i^{max}}\right)^2 \\ \text{subject to } & \tau(j) = R_m f_m(a(j)) + \beta(j), \\ & 0 \le a_i(j) \le 1, \; \forall i \in [1,\ldots,m] \\ & -\beta_i^{max} \le \beta_i(j) \le \beta_i^{max}, \; \forall i \in [1,\ldots,n] \end{aligned} \tag{43}$$

where $\beta \in \Re^n$ are the complementary residual activations, γ is the penalty constant and β_i^{max} is the maximum allowed value for the residual force i. Although the introduction of residuals does not have a direct physiological counterpart, their use allows for investigation of model weaknesses. The process of choosing γ and β^{max} is as follows: 1) assign high values to β^{max}, $\gamma = 1$ and perform a first pass simulation, 2) update $\beta_i^{max} = max(\beta_i(t = start : end))\forall i$ and 3) experimentally sample $\gamma \in [0, \infty)$ in order to assess whether residual forces are further reduced. We adopt an interior point method [46] with constraint and convergence tolerance of 10^{-5}. A final note is that the dynamic redundancy is handled in a forward manner [26] (e.g. the muscles are activated and the developed forces are compared against the required generalized forces).

III. Results

In Subsection III-A we present a comparison between the two alternative constrained models (A and B) derived in Subsection II-C. More specifically, we show that the choice of the constrained model can alter the required generalized forces without altering the motion, in the sense that the constraints can be used to reduce the command applied by the controller. Moreover, we demonstrate how to plan a movement behavior in task space and perform assessments without any experimental measurements.

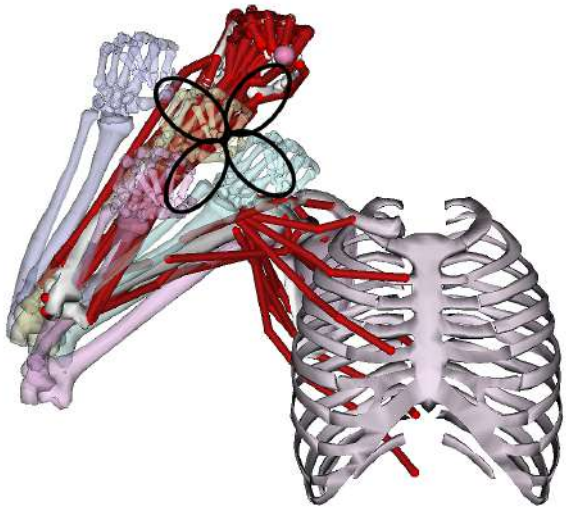


Fig. 4: This figure depicts the time instances of a customly planned trajectory. The upper extremity model [47] is comprised of seven DoFs and forty seven muscles. The model defines a set of constraint equations that regulate the movement of the shoulder complex.

For this particular experiment the upper extremity model [47] is used (Fig. 4). It consists of seven DoFs, including shoulder rotation and elevation (thoracohumeral angle), wrist flexion, wrist deviation, elbow flexion, elevation plane of the shoulder and forearm rotation. There are forty seven muscles (Schutte muscle model [48]) spanning these joints. More importantly, it defines the movement of the shoulder complex through a set of constraints, making it a good candidate to evaluate the constrained models.

Next, we demonstrate that the two presented algorithms can be employed to solve the ID problem. A complex behavior of a gait movement along with the ground reaction forces recordings is used as a benchmark. For the IK problem, the marker error metric is adopted as a baseline for comparison, where

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

we show that despite the nature of the proposed algorithm (TSDIK), its tracking error is relatively low compared to the optimization-based approach (Subsection III-B). Finally, we estimate the required muscle patterns through TSCMC and compare the results against CMC algorithm (Subsection III-C).

The gait2354 model (Fig. 2 shown without muscles), which has twenty three DoFs and fifty four muscles (Thelen muscle model [20]), is used for the second experiment. The model features lower extremity joint definitions adapted from [49], low back joint and anthropometry from [50] and a planar knee model from [51].

A. Comparison of the Constrained Models

We define the following trajectory to demonstrate the task space planning mechanism

$$\theta = \pi t, r = Asin(2\theta)$$
$$x(t) = x(0), y(t) = y(0) + rsin(\theta), z(t) = z(0) + rcos(\theta) \quad (44)$$

which generates the 3D movement presented in Fig. 4. We can assign a position task on the hand and prescribe the desired trajectory following Eq. (44). The muscle excitation patterns and other variables of interest can be assessed utilizing the TSCMC algorithm. Here we compare the two constrained models from equations (16) and (19) with respect to the simulated generalized forces.

The simulated trajectory and generalized coordinates are shown in Fig. 5 and Fig. S-8, respectively. It is evident by these figures that the same movement is generated by the two models, both in terms of goal trajectory and model coordinates, something that is in good agreement with the motion-constraint separation principle (Subsection II-C).

It is worth noting that the two models define different projection operators, meaning they issue different joint space forces despite resulting in identical movement behaviors. From the derivation, the total generalized forces is the sum of the motion related component (task and null space forces Eq. (31)). Fig. 6 depicts the magnitude ($||\tau||$) of each component for the two alternative models. Model-A requires larger motion forces compared to Model-B, although Model-A generates smaller constraint forces. This suggests that Model-B utilized the constraint forces more rigorously, minimizing the required motion forces ($\tau^T\tau$), while Model-A minimizes the inertia weighted command ($\tau^T M^{-1}\tau$) [34]. Conclusively, the inverse model can impact the required generalized forces for tracking a movement behavior and consequently the muscle effort.

The fact that the two models produce different forces, but generate identical movement implies that this particular model is overconstrained. Let's assume that a model contains n unconstrained DoFs, c constraint algebraic equations and n_a active DoFs. Then the actual DoFs of the constrained model are of dimension $n_c = n - c$. We can distinguish three cases ([34]):

1) $n_c > n_a$, the system is underactuated. There is at most one solution to the ID problem.
2) $n_c = n_a$, the system is fully actuated. There is a unique solution.
3) $n_c < n_a$, the system is overconstrained. There are an infinite number of solutions (τ) that will achieve the desired goals (this is the case here).

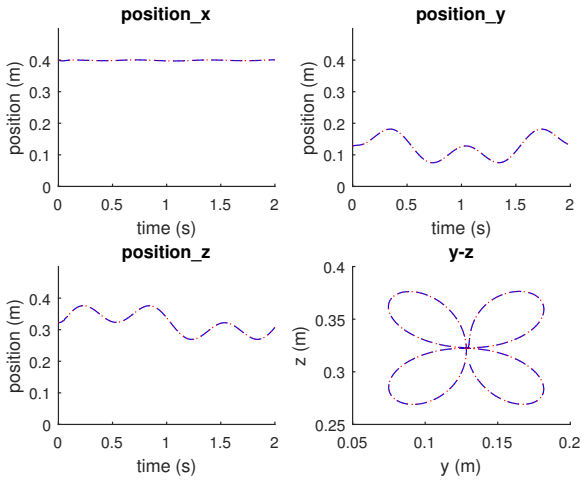


Fig. 5: The simulated task trajectory of the upper limb model for the finger point task. The $y-z$ plane defines the resulting pattern. The trajectories of Model-A (Eq. (16)) are drawn with red dotted lines, while the blue dashed lines denote the trajectories of Model-B (Eq. (19)). Both models result in identical motion.

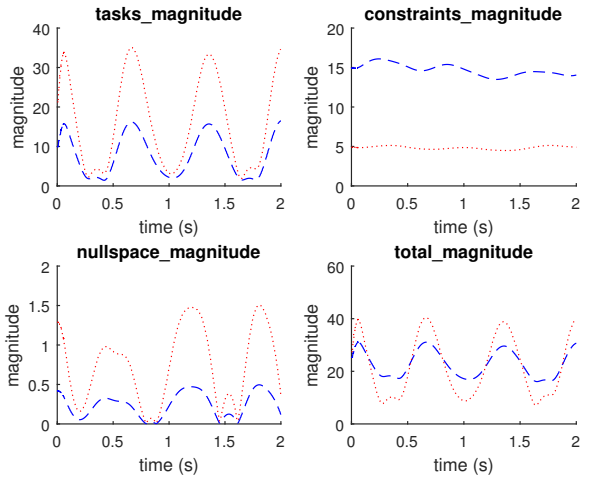


Fig. 6: The computed forces that drive the model to perform the required movement. These forces are decoupled into motion related forces (task and null space forces) and constraint forces. A comparison of the magnitude values for the two models is presented. The trajectories of Model-A (Eq. (16)) are drawn with red dotted lines, while the blue dashed lines denote the trajectories of Model-B (Eq. (19)). Model-A results in larger motion forces command compared to Model-B, while Model-B results in larger constraint forces than Model-A.

B. Inverse Kinematics on Gait

The evaluation of the IK algorithm is presented for a single gait cycle. A comparison between the available optimization-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

based algorithm (OpenSim) and the TSDIK is shown in Fig. 7, in terms of the total marker RMS, min and max error. In general, the optimization-based algorithm behaves better in terms of tracking. Despite the nature of the presented algorithm (TSDIK), its tracking error is relatively low and acceptable ($< 2cm$ total RMS).

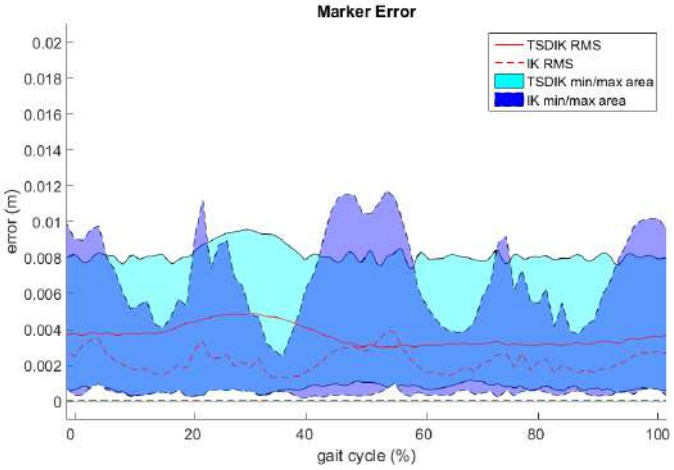


Fig. 7: Comparison of the total RMS, min and max error between the TSDIK and the optimization-based IK algorithms for a single gait cycle. The shaded area encloses the min and max marker error as its upper and lower bound respectively.

As far as simulation performance is concerned our approach is limited by its dynamic nature. More specifically, the explicit numerical integration scheme and the large accelerations required to track the movement takes up about 95% of the total execution time, as indicated by profiling (Subsection S-F). Implicit integration schemes are not available in the current version of OpenSim, nevertheless it has been shown that they can improve the execution time dramatically [52].

In the case of the optimization-based algorithms the weight of each error term contributed by each marker can be controlled separately, while the proposed algorithm relies on the task prioritization. The priority graph used for the gait movement follows the tree structure of the multibody system (high priority for the proximal segments and lower priority for the distal). It can be argued that this kind of priority structure is dependent on the type of movement that is analyzed, as well as the presence of uncertainty in the measurements. For example, in gait related movements the floating base should have a higher priority, as was observed during simulation. Otherwise, in the case of a reach movement the hand tracking should be prioritized higher than the more proximal segments. Therefore, the user has to decide on the priority graph with respect to the movement under study.

C. Inverse Muscle Driven Simulation on Gait

A comparison of the TSCMC with the CMC method for the same gait movement was performed. The task goals generated by the TSDIK algorithm were supplied to the TSCMC method and similarly the output movement of the optimization-based IK was used by CMC. Fig. S-9 depicts the computed controls of the two algorithms. The estimated muscle patterns show very good agreement between the two methods, despite their different internal implementations. The execution time performance of the two algorithms was approximately the same (differences $0.006\% \pm 0.001\%$).

The residual forces/torques as computed by the two algorithms are shown in Fig. S-10. The goal is to keep these residual as low as possible. Major differences are present in the residual forces, where for some coordinates the proposed algorithm performs better and vice versa. The Residual Reduction Algorithm (RRA) [17], [53] method has not been applied to reduce the residuals prior to the simulation, since we didn't want to bias the comparison between the two methods. We can conclude that the proposed method is generally equivalent to CMC in terms of assessing the muscle effort distribution.

IV. Discussion

Although joint space representation is the de facto standard for formulating the underlying EoM and dynamics simulation methods, it may be suboptimal for a certain range of problems. Common movement behaviors can be described through a well defined set of interrelating task goals, which is especially evident in the process of planning a movement in the task space. Projecting the EoM in the domain of the task provides a straightforward mechanism to translate a movement behavior to the biomechanical counterpart. The underlying complexity of mapping the abstract task primitive to a movement is effectively handled by the underlying controllers. This work mainly focuses on motion primitive tasks, without necessarily restricting applications to those alone [39]. Extension to other types of tasks can shed some light towards understanding the complex interactions between the nervous system and the execution of multiple task goals.

The underlying controller (ITSC) was designed to handle constrained multibody systems, facilitating the study of broader types of problems. It is straightforward to verify that constrained task space projection is coordinate invariant, in the sense that the provided algorithms can address systems that use absolute coordinates [12]. This is true since the underlying projection operators are formed from the fundamental system quantities (e.g. the inertial mass matrix, the constraint and task Jacobian) that can be derived under any coordinate system. Another virtue of constraint modeling is that the presented controllers can address systems of closed kinematic chains, since they are commonly modeled by constraints as a consequence of the virtual kinematic chain principle [54], [55]. It is important to note that closed kinematic chains are found to be the rule in many biological systems rather than the exception [56]. As constraints can be used to model biological systems more closely, results demonstrate that they can alter the required muscle forces. Consequently, the effect of constraint modeling on the assessment of muscle forces remains a subject open to study.

As a proof of concept we showed that task-based schemes can be used to solve the IK problem. Experimental evaluation indicates an acceptably low tracking error for a complex gait movement. The execution time of the algorithm is inferior

as compared to the optimization-based approach, due to a combination of an explicit numerical integration scheme and the high acceleration goals required to track the movement. The authors of [52] suggested that implicit numerical integration can dramatically improve the execution time of the FD methods. Dynamics-based IK approaches can be of great interest, since constraints are primarily resolved at a dynamic level as opposed to kinematic methods, they can be paired with temporal filtering techniques for reducing marker error artifacts ([57]) and allow for a combination of different types of input (e.g. IMU and marker recordings).

We showed that the TSCMC algorithm is able to establish a bilateral link between the task(s) goals and muscle excitation patterns. Moreover, we compared its performance to the CMC algorithm and demonstrated negligible differences. The algorithm can be used in combination with a high level controller that generates a set of abstract task goals in order to perform simulation prior to any experimental data collection. This is of great importance, since simulations can be arranged effortlessly and intuitively. Of equal importance is the fact that our method allows to deduce important variables that can be used to improve the underlying model and the experimental setup a priori. Conclusively, it offers a novel, abstract point of view and control, which can prove to be advantageous towards further integration with high level models of the precommand level [1], [58].

V. Conclusion

We presented a set of algorithms for solving the inverse and forward dynamics problem in task space for constrained musculoskeletal systems. Task space projection provides an alternative representation over joint space coordinates, leading to more intuitive and straightforward motion action planning. The evaluation of the proposed algorithms shows good overall performance compared to the state of the art methods. Detailed derivations were presented in order to provide the reader with the appropriate background so as to understand the implications of constraint modeling and task space projection. After highlighting some important implications and limitations of this work, we conclude that the task-based approach could be adopted in the design of simulations related to the study of constrained musculoskeletal systems, especially towards higher level integration with brain models.

Acknowledgment

We would like to thank the OpenSim team and especially Christopher Dembia and Michael Sherman for their constructive suggestions on technical issues regarding the implementation of the proposed framework. Additionally, we would like to thank Samir Menon on his comments on the operation task space formalization and the DMC algorithm. Finally, we would like to thank Aristotelis Spathis-Papadiotis for his valuable remarks.

References

[1] T. DeWolf and C Eliasmith, "The neural optimal control hierarchy for motor control," *Journal of Neural Engineering*, vol. 8, no. 6, pp. 1–21, 2011.

[2] D. M. Wolpert, "Computational approaches to motor control," *Nature*, vol. 1, no. 6, pp. 209–216, 1997.

[3] F. J. Valero-Cuevas, H. Hoffmann, *et al.*, "Computational models for neuromuscular function," *IEEE Rev Biomed Eng*, vol. 2, pp. 110–135, 2009.

[4] F. J. Valero-Cuevas, "A mathematical approach to the mechanical capabilities of limbs and fingers," *Adv Exo Med Biol*, vol. 629, pp. 619–633, 2009.

[5] J. J. Kutch and F. J. Valero-Cuevas, "Challenges and new approaches to proving the existence of muscle synergies of neural origin," *PLoS Computational Biology*, vol. 8, no. 5, pp. 1–11, 2012.

[6] J. M. Inouye and F. J. Valero-Cuevas, "Muscle synergies heavily influence the neural control of arm endpoint stiffness and energy consumption," *PLoS Computational Biology*, vol. 12, no. 2, pp. 1–24, 2016.

[7] S. Davoudabadi Farahani, M. Svinin, *et al.*, "Prediction of closed-chain human arm dynamics in a crank-rotation task," *Journal of Biomechanics*, vol. 49, no. 13, pp. 2684–2693, 2016.

[8] M. Kawato, "Internal models for motor control and trajectory planning," *Current Opinion in Neurobiology*, vol. 9, no. 6, pp. 718–727, 1999.

[9] N. Hogan, "Adaptive control of mechanical impedance by coactivation of antagonist muscles," *IEEE Transactions on Automatic Control*, vol. 29, no. 8, pp. 681–690, 1984.

[10] O. Khatib, "Inertial properties in robotic manipulation: an object-level framework," *International Journal of Robotics Research*, vol. 14, no. 1, pp. 19–36, 1995.

[11] A. Seth, M. A. Sherman, *et al.*, "Minimal formulation of joint motion for biomechanisms," *Nonlinear Dynamics*, vol. 62, no. 1-2, pp. 291–303, 2010.

[12] K. Dziewiecki, W. Blajer, *et al.*, "Modeling and computational issues in the inverse dynamics simulation of triple jump," *Multibody System Dynamics*, vol. 32, no. 3, pp. 299–316, 2014.

[13] V. De Sapio and J. Park, "Multitask constrained motion control using a mass-weighted orthogonal decomposition," *Journal of Applied Mechanics*, vol. 77, no. 4, pp. 1–10, 2010.

[14] O. Khatib, E. Demircan, *et al.*, "Robotics-based synthesis of human motion," *Journal of Physiology-Paris*, vol. 103, no. 3-5, pp. 211–219, 2009.

[15] A. Erdemir, S. G. McLean, *et al.*, "Model-based estimation of muscle forces exerted during movements," *Clinical Biomechanics*, vol. 22, no. 2, pp. 131–154, 2007.

[16] D. G. Thelen and F. C. Anderson, "Using computed muscle control to generate forward dynamic simulations of human walking from experimental data.," *Journal of Biomechanics*, vol. 39, no. 6, pp. 1107–15, 2006.

[17] S. L. Delp, F. C. Anderson, *et al.*, "Opensim : open-source software to create and analyze dynamic simulations of movement," *IEEE Transactions on Biomedical Engineering*, vol. 54, no. 11, pp. 1940–1950, 2007.

[18] A. Erdemir, T. M. Guess, *et al.*, "Commentary on the integration of model sharing and reproducibility analysis to scholarly publishing workflow in computational biomechanics," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 10, pp. 2080–2085, 2016.

[19] O. Khatib, L. Sentis, *et al.*, "Whole-body dynamic behavior and control of human-like robots," *International Journal of Humanoid Robotics*, vol. 1, no. 1, pp. 29–43, 2004.

[20] D. G. Thelen, "Adjustment of muscle mechanics model parameters to simulate dynamic contractions in older adults," *Journal of Biomechanical Engineering*, vol. 125, no. 1, pp. 70–77, 2003.

[21] M. Millard, T. Uchida, *et al.*, "Flexing computational muscle: modeling and simulation of musculotendon dynamics," *Journal of Bomechanical Engineering*, vol. 135, no. 2, pp. 1–12, 2013.

[22] C. A. Myers, K. B. Shelburne, *et al.*, "A probabilistic approach to quantify the impact of uncertainty propagation in musculoskeletal simulations," *Annals of Biomedical Engineering*, vol. 43, no. 5, pp. 1098–1111, 2015.

[23] A. S. Arnold, D. J. Asakawa, *et al.*, "Do the hamstrings and adductors contribute to excessive internal rotation of the hip in persons with cerebral palsy?" *Gait and Posture*, vol. 11, no. 3, pp. 181–190, 2000.

[24] M. G. Pandy, "Computer modeling and simulation of human movement," *Annals of Biomedical Engineering*, vol. 3, pp. 245–73, 2001.

[25] T. S. Buchanan, D. G. Lloyd, *et al.*, "Neuromusculoskeletal modeling: estimation of muscle forces and joint moments and movements from measurements of neural command," *Journal of Applied Biomechanics*, vol. 20, no. 4, pp. 367–395, 2006.

[26] M. S. Shourijeh, K. B. Smale, *et al.*, "A forward-muscular inverse-skeletal dynamics framework for human musculoskeletal simulations," *Journal of Biomechanics*, vol. 49, no. 9, pp. 1718–1723, 2016.

[27] C. Ong, J. Hicks, *et al.*, "Simulation-based design for wearable robotic systems: an optimization framework for enhancing a standing long jump," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 5, pp. 894–903, 2016.

[28] C. Pizzolato, D. G. Lloyd, *et al.*, "Ceinms: a toolbox to investigate the influence of different neural control solutions on the prediction of muscle excitation and joint moments during dynamic motor tasks," *Journal of Biomechanics*, vol. 48, no. 14, pp. 3929–3936, 2015.

[29] G. Durandau, D. Farina, *et al.*, "Real-time musculoskeletal modeling driven by electromyograms," *IEEE Transactions on Biomedical Engineering*, vol. PP, no. 99, pp. 1–8, 2017.

[30] J. L. Dideriksen, R. M. Enoka, *et al.*, "A model of the surface electromyogram in pathological tremor," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 8, pp. 2178–2185, 2011.

[31] R Mootanah, C. W. Imhauser, *et al.*, "Development and validation of a computational model of the knee joint for the evaluation of surgical treatments for osteoarthritis," *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 17, no. 13, pp. 1502–17, 2014.

[32] E. K. Chadwick, D. Blana, *et al.*, "Real-time simulation of three-dimensional shoulder girdle and arm dynamics," *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 7, pp. 1947–1956, 2014.

[33] F. Aghili, "A unified approach for inverse and direct dynamics of constrained multibody systems based on linear projection operator: applications to control and simulation," *IEEE Transactions on Robotics*, vol. 21, no. 5, pp. 834–849, 2005.

[34] L. Righetti, J. Buchli, *et al.*, "Inverse dynamics control of floating-base robots with external contraints: an unified view," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 1085–1090.

[35] M. Mistry and L. Righetti, "Operational space control of constrained and underactuated systems," in *Proceedings of Robotics: Science and Systems*, 2011, pp. 225 –232.

[36] E. Eich, "Convergence results for a coordinate projection method applied to mechanical systems with algebraic constraints," *SIAM Journal on Numerical Analysis*, vol. 30, no. 5, pp. 1467–1482, 1993.

[37] J. Baumgarte, "Stabilization of constraints and integrals of motion in dynamical systems," *Computer Methods in Applied Mechanics and Engineering*, vol. 1, no. 1, pp. 1–16, 1972.

[38] W. Fisher and S. Mujtaba, "Hybrid position/force control: a correct formulation," *The International Journal of Robotics Research*, vol. 11, no. 4, pp. 299–311, 1991.

[39] L. Sentis, *Synthesis and Control of Whole-Body Behaviors in Humanoid Systems*. Stanford University, 2007, pp. 1–208.

[40] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America A*, vol. 4, no. 4, pp. 629–642, 1987.

[41] G. M. Nielson, "V-quaternion splines for the smooth interpolation of orientations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 10, no. 2, pp. 224–229, 2004.

[42] F. C. Anderson and M. G. Pandy, "Static and dynamic optimization solutions for gait are practically equivalent," *Journal of Biomechanics*, vol. 34, no. 2, pp. 153–161, 2001.

[43] M. Sherman, A. Seth, *et al.*, "What is a moment arm? calculating muscle effectiveness in biomechanical models using generalized coordinates," in *Proc ASME Des Eng Tech Conf.*, 2013, pp. 1–18.

[44] A. D. Deshpande, R. Balasubramanian, *et al.*, "Acquiring variable moment arms for index finger using a robotic testbed," *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 8, pp. 2034–2044, 2010.

[45] F. E. Zajac, "Muscle and tendon: properties, models, scaling and application to biomechanics and motor control," *Critical Reviews in Biomedical Engineering*, vol. 17, no. 4, pp. 359–411, 1989.

[46] A. Wachter and L. T. Biegler, *On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming*, 1. 2006, vol. 106, pp. 25–57.

[47] K. R. Saul, X. Hu, *et al.*, "Benchmarking of dynamic simulation predictions in two software platforms using an upper limb musculoskeletal model.," *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 18, no. 13, pp. 1–14, 2014.

[48] L. M. Schutte, M. M. Rodgers, *et al.*, "Improving the efficacy of electrical stimulation-induced leg cycle ergometry: an analysis based on a dynamic musculoskeletal model," *IEEE Transactions on Rehabilitation Engineering*, vol. 1, no. 2, pp. 109–125, 1993.

[49] S. L. Delp, P. J. Loan, *et al.*, "An interactive graphics-based model of the lower extremity to study orthopedic surgical procedures," *IEEE Transactions on Biomedical Engineering*, vol. 37, no. 8, pp. 757–767, 1990.

[50] F. C. Anderson and M. G. Pandy, "A dynamic optimization solution for vertical jumping in three dimensions," *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 2, no. 3, pp. 201–231, 1999.

[51] G. T. Yamaguchi and F. E. Zajac, "A planar model of the knee joint to characterize the knee extensor mechanism," *Journal of Bomechanics*, vol. 22, no. 1, pp. 1–10, 1989.

[52] A. J. van den Bogert, D. Blana, *et al.*, "Implicit methods for efficient musculoskeletal simulation and optimal control," *Procedia IUTAM*, vol. 2, no. 2011, pp. 297–316, 2011.

[53] S. R. Hamner, A. Seth, *et al.*, "Muscle contributions to propulsion and support during running," *Journal of Biomechanics*, vol. 43, no. 14, pp. 2709–16, 2010.

[54] A. Campos, R. Guenther, *et al.*, "Differential kinematics of serial manipulators using virtual chains," *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, vol. 27, no. 4, pp. 345–356, 2005.

[55] A. Jain, *Robot and Multibody Dynamics*. Springer, 2011, pp. 1–510.

[56] S. Levin, S. L. de Solorzano, *et al.*, "The significance of closed kinematic chains to biological movement and dynamic stability," *Journal of Bodywork and Movement Therapies*, vol. 21, no. 3, pp. 664–672, 2017.

[57] F. De Groote, T. De Laet, *et al.*, "Kalman smoothing improves the estimation of joint kinematics and kinetics in marker-based human gait analysis," *Journal of Biomechanics*, vol. 41, no. 16, pp. 3390–3398, 2008.

[58] D. Zhang, P. Poignet, *et al.*, "Exploring peripheral mechanism of tremor on neuromusculoskeletal model: a general simulation study.," *IEEE Transactions on Biomedical Engineering*, vol. 56, no. 10, pp. 2359–69, 2009.

[59] J. Nakanishi, R. Cory, *et al.*, "Operational space control: a theoretical and empirical comparison," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 737 –757, 2008.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

SUPPLEMENTARY MATERIAL

A. SimTK Project

The source code along with any related material for this publication can be found at the following SimTK project: https://simtk.org/projects/task-space.

B. Kabsch Algorithm

Algorithm S-2 Kabsch algorithm for finding the optimal rotation and translation between two corresponding 3D point clouds.

Input: ${}^{G}p^{b}_{:}(n), {}^{G}p^{b}_{:}(n+1)$
Output: ${}^{G}T^{b}(n, n+1) = \{R, t\}, R \in \Re^{3\times3}, t \in \Re^{3}$

1: Find centroid of each point set c_A, c_B
   $c = \frac{1}{N}\sum_{i=1}^{N} {}^{G}p^{b}_{i}$
2: Compute the cross covariance matrix
   $H = \sum_{i=1}^{N}({}^{G}p^{b}_{i}(n) - c_A)({}^{G}p^{b}_{i}(n+1) - c_B)^T$
3: Perform singular value decomposition on H
   $[U, S, V] = svd(H)$
4: Compute the rotation matrix
   $R = VU^T$
   Handle reflexion case
5: **if** $(def(R) < 0)$ **then**
6: multiply 3rd column of R by -1
7: **end if**
8: Find the translation
   $t = -R \cdot c_A + c_B$
9: **return** $\{R, t\}$

C. Results

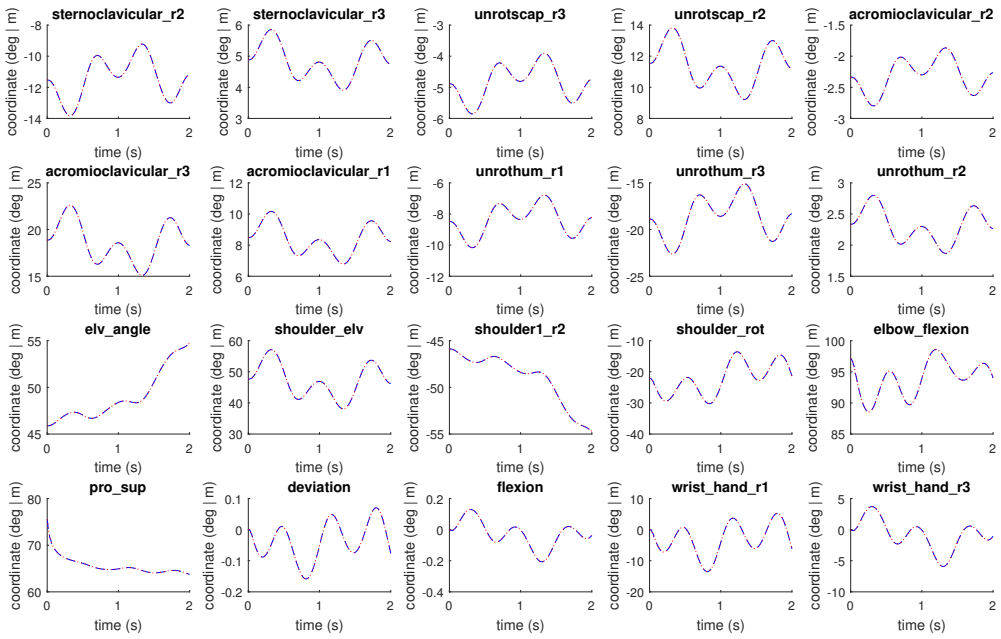


Fig. S-8: The simulated generalized coordinates (q) of the upper limb model as a result for the defined custom trajectory. The trajectories of Model-A (Eq. (16)) are drawn with red dotted lines, while the blue dashed lines denote the trajectories of Model-B (Eq. (19)). Both models result in identical motion (residuals are zero).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

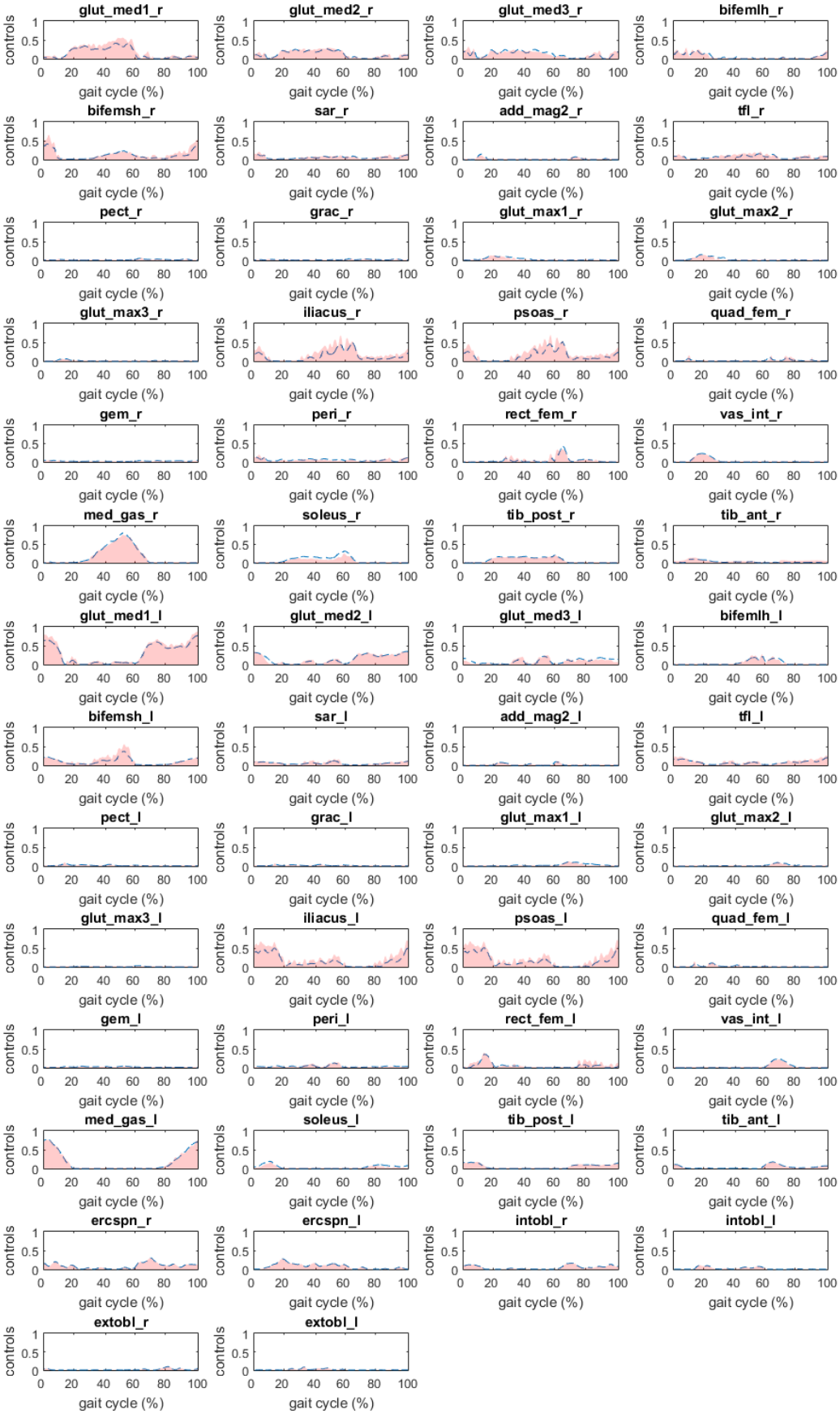


Fig. S-9: This figure presents the muscles excitations for a single gait cycle in comparison between CMC (red solid line) method and the proposed TSCMC (blue dashed line). There are some minor differences, as the two methods have their implementation specifics, but in general they agree.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

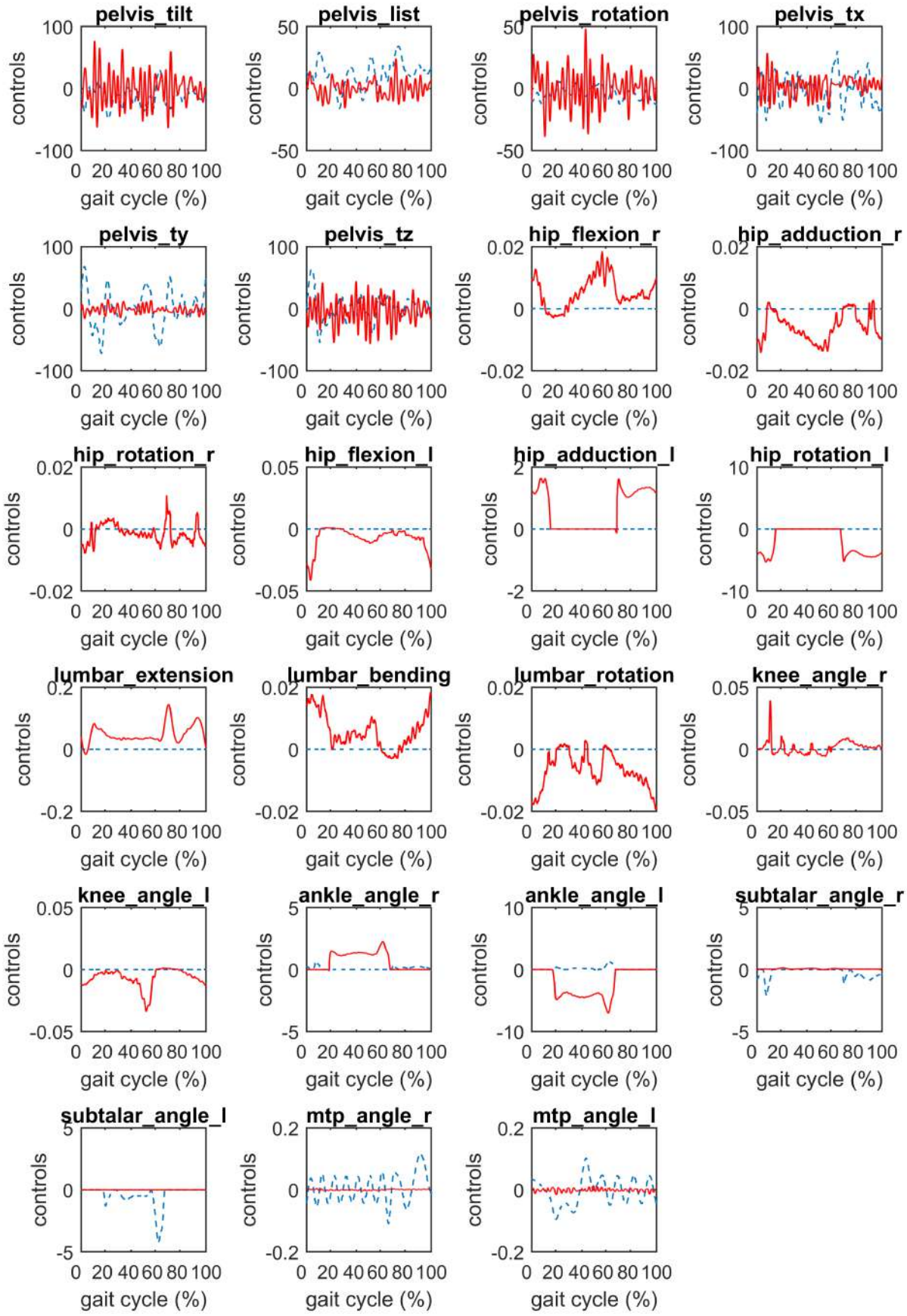
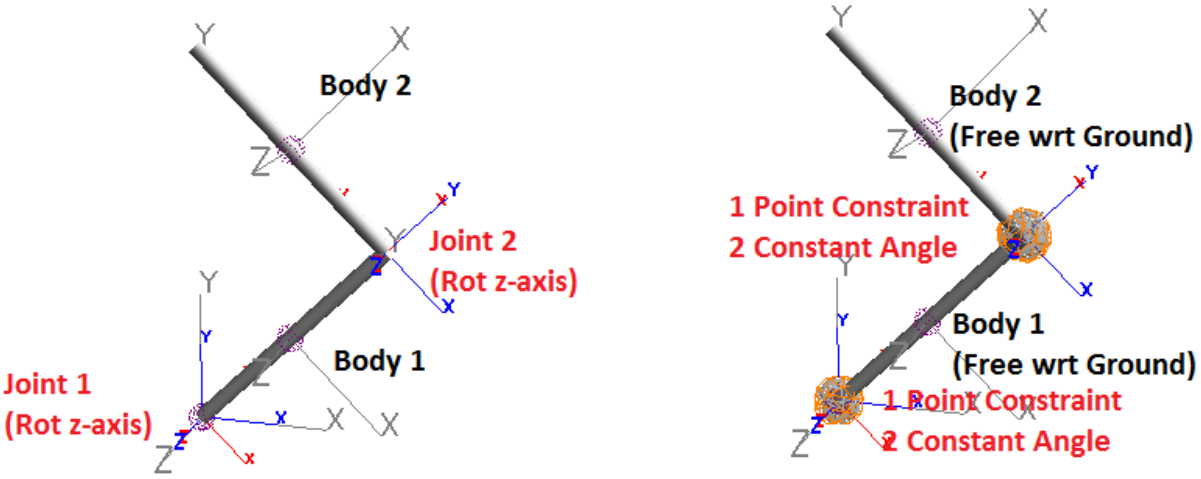


Fig. S-10: This figure presents the residual forces/torques for a single gait cycle in comparison between CMC (red solid line) method and the proposed TSCMC (blue dashed line).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

D. Absolute (Cartesian) Coordinates



(a) The unconstrained model is composed of two bodies and two pin joints (1 and 2). A pin joint permits a rotational movement around an axis (in this case is the z-axis).

(b) The free body model is composed of two bodies that are permitted to move freely with respect to the ground frame (absolute coordinates are used). To enforce similar functional behavior with respect to the model on the left, the bodies are further constrained at the joint level by a point constraint and two constant angle constraints (for each joint).

Fig. S-11: Left: an unconstrained model (generalized coordinates) with two DoFs. Right: constrained model that uses absolute coordinates twelve DoFs and ten constraint algebraic equations.

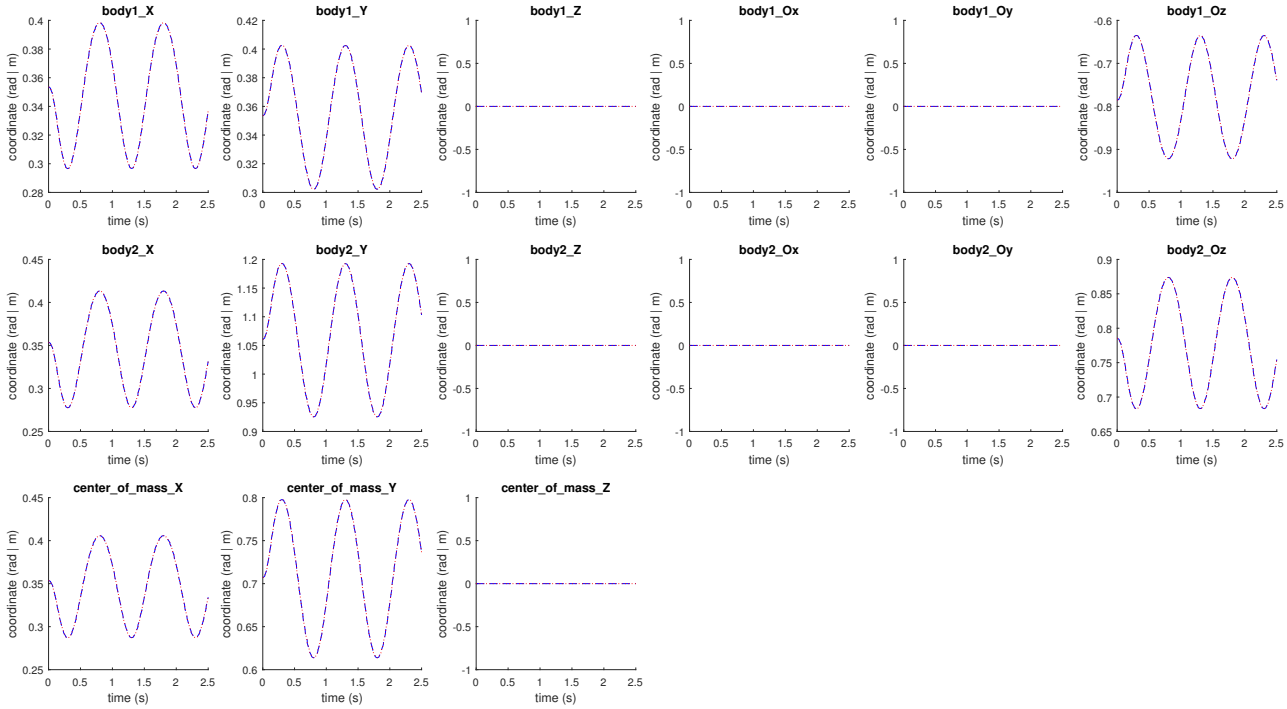


Fig. S-12: Body kinematics analysis of the two DoF (generalized coordinates) (blue) and the model that uses absolute coordinate (red). XYZ quantities correspond to the translational components, while $Oxyz$ quantities represent the orientation of the bodies. The task goal is a sinusoidal movement of the tip of the second body in the y direction. The two models produce identical movements (residuals are zero).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

In order to evaluate whether this framework can be used to simulate models that use absolute (Cartesian) coordinates, we introduce two identically functional models Fig. S-11. The model on the left (Fig. S-11a) consists of two bodies and two DoFs (generalized coordinates are used). The first and second bodies are permitted to rotate around the z-axis by a pin joint (one rotation DoF). The model on the right (Fig. S-11b) treats each body as a free body that is permitted to move freely with respect to the ground frame. Furthermore, the first body is constrained to the ground by a point constraint, which restricts relative translation of a point on the two bodies (three algebraic equations), and two constant angle constraints for restricting rotation around x and y axis (two algebraic equations). Similarly, the second body is constrained with respect to the first body. This model contains twelve DoFs and ten constraint algebraic equations. In terms of permissible movements behaves identically to the first model.

The goal is to use the task driven controller (ITSC) to drive the tip of second body in performing a sinusoidal movement in the y direction

$$x_d(t) = [x_x(0), x_y(0) + Asin(2\pi t), x_z(0)]^T \tag{S-45}$$

where $x_d(t)$ is the desired position at time t and $A = 0.2m$.

Results show that the ITSC is able to handle systems that use absolute coordinates. Fig. S-12 demonstrates that the two models (Fig. S-11) produce identical movement. We conclude that the presented task space controller is coordinate invariant as long as the fundamental quantities of the model (e.g. inertia mass matrix, Coriolis and centrifugal forces, gravity forces, constraint Jacobian, body Jacobian, etc.) are available.

E. Simulation of Closed Kinematic Chains

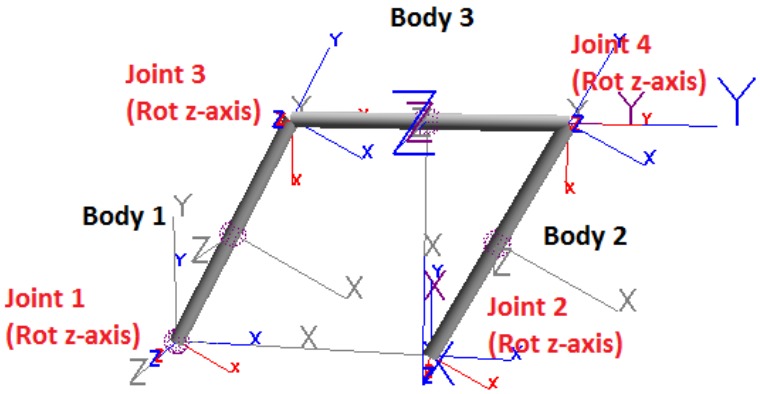


Fig. S-13: A model of a closed kinematic chain topology. The first body (left) is permitted to rotate around the z-axis with respect to the ground frame. The second body (right) is connected to the ground by an offset of the length of the third body (top) and it is also permitted to rotate around the z-axis. The third body is connected with the first and second bodies by two pin joints.

Closed kinematic chains are commonly handled [55] by cutting the kinematic chain and introducing a constraint that connects the two (the virtual kinematic chain principle [54]). When generalized coordinates are used the closed chain can be separated by cutting a body instead of a joint, so that the original coordinates of the model are preserved. When absolute coordinates are used the closed chain can be segmented at the joint level. By doing so instead of having a closed chain two separate open chains are constructed, that are appropriately constrained. As a consequence, a controller that accounts for the constraint forces can handle closed kinematic chains.

As presented in Fig. S-13 the model consist of three bodies that are permitted to rotate around the z-axis. The top most body is connected by two joints with the two supporting bodies (left, right). The underlying multibody dynamics engine (Simbody) implicitly transforms closed kinematic chains by cutting a body rather than a joint. Mass properties are divided between the two halves. Then a weld constraint (six algebraic equations) is used to reattach the two halves.

To prove that the proposed controller (ITSC) is capable of controlling this system an orientation task is assigned to the leftmost body (body 1)

$$\theta_d(t) = [\theta_x(0), \theta_y(0), \theta_z(0) + \frac{\pi}{2} sin(2\pi t)]^T \tag{S-46}$$

where $\theta_d(t)$ is the desired orientation at time t. Fig. S-14 depicts the simulated body orientations, which validate the correct behavior of the controller. Therefore the proposed controller is capable of addressing topologies that contain closed kinematic chains.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

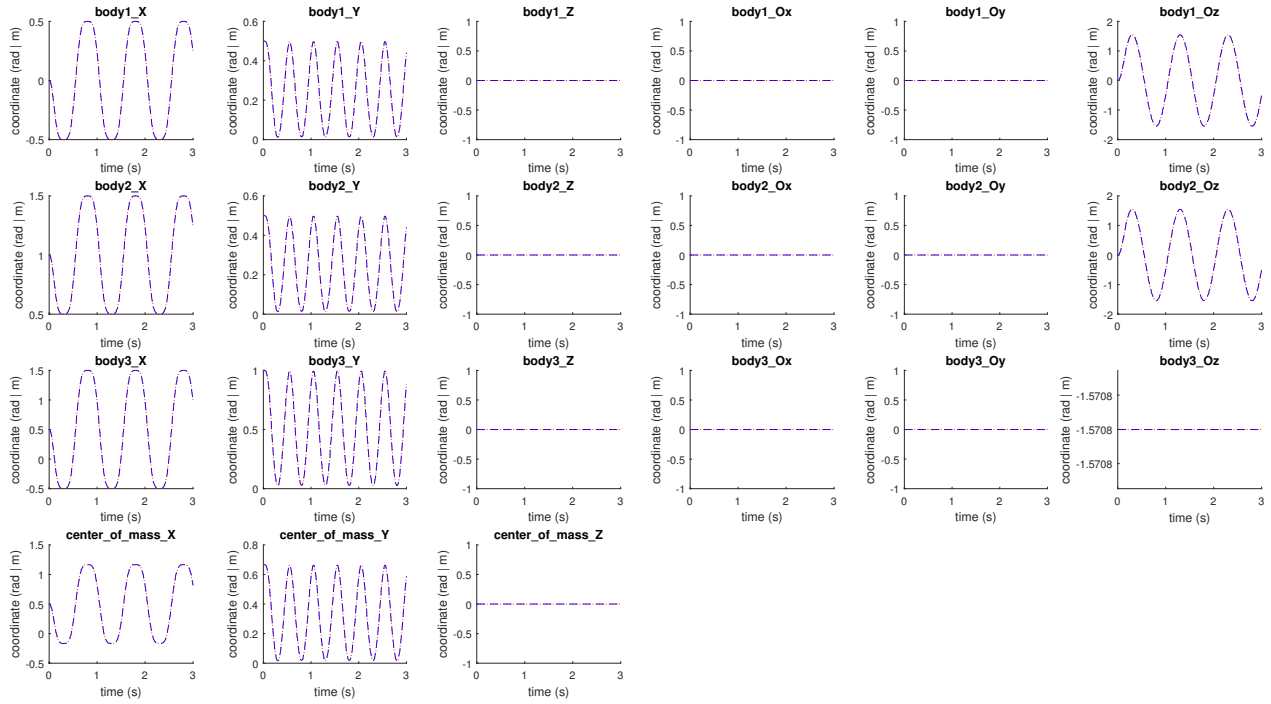


Fig. S-14: Body kinematics analysis of the two models Model-A (blue), Model-B (red) on a closed kinematic chain model. XYZ quantities correspond to the translational components, while $Oxyz$ quantities represent the orientation of the bodies. The goal is to rotate body 1 along the z-axis. Since body 2 is connected to body 1 by body 3 (top) their orientation is identical. Furthermore, because of the structure we expect that body 3 do not rotate at all for this particular experiment. The two models produce identical movements (residuals are zero).

F. Algorithm Performance Analysis

The following analysis was performed on Windows 10, Intel Core i7-4771, 3.5GHz, 8GB RAM, SanDisk SSD disk device. Table S-1 presents the execution time of the proposed TSDIK and the optimization-based IK algorithms. The latter clearly outperforms the dynamics-based algorithm. We investigated the cause of this difference and found that 95% of the execution time (Fig. S-15) is spent on integrating the system. Good tracking requires large task accelerations that consequently enforce the integrator in taking smaller time steps. To validate this we changed the PD control law of the tracking controller

$$a(t) = a_d(t) + 200(x_d(t) - x(t)) + 20(u_d(t) - u(t)) \tag{S-47}$$

$$a(t) = 0 \cdot a_d(t) + 200(x_d(t) - x(t)) + 20(u_d(t) - u(t)) \tag{S-48}$$

In the first experiment we used all available information (Table S-1, TSDIK-1) and for the second we ignored the acceleration term (Table S-1, TSDIK-2). The tracking error of the second experiment was worse (max RMS 0.046m compared to 0.005m), since the acceleration information was not used (similar remarks, e.g. between velocity and acceleration-based approaches were mentioned in [59]). The authors of [52] claim that implicit numerical integration can improve the execution time. Unfortunately, our implementation platform (OpenSim/Simbody) does not provide support for implicit integration. Please note that the proposed algorithms have not been optimized for speed, where the interior point method [46] used in OpenSim/Simbody is well implemented and properly optimized for performance.

TABLE S-1: Execution time of the IK and TSDIK methods as a function of the number of bodies. TSDIK-1 uses the acceleration information in the PD controller, while TSDIK-2 does not.

Bodies (#)	IK (sec.)	TSDIK-1 (sec.)	TSDIK-2 (sec.)
1	0.013	0.57	0.22
2	0.023	1.14	0.53
3	0.041	1.70	0.62
4	0.069	3.10	1.14
12	0.24	15.744	5.10

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

Comparison between CMC and TSCMC did not show any significant differences in the execution time. This was expected as the two approaches are very similar, e.g. both use tracking controllers in combination with an optimization scheme in a FD manner. Furthermore, if ideal muscles are used the execution times reduces dramatically (from approximately 50min to 3min). This suggests that muscle dynamics make the system stiff. On the other hand SO is significantly faster compared to both algorithms [42], since it is a static method that does not use numerical integration.

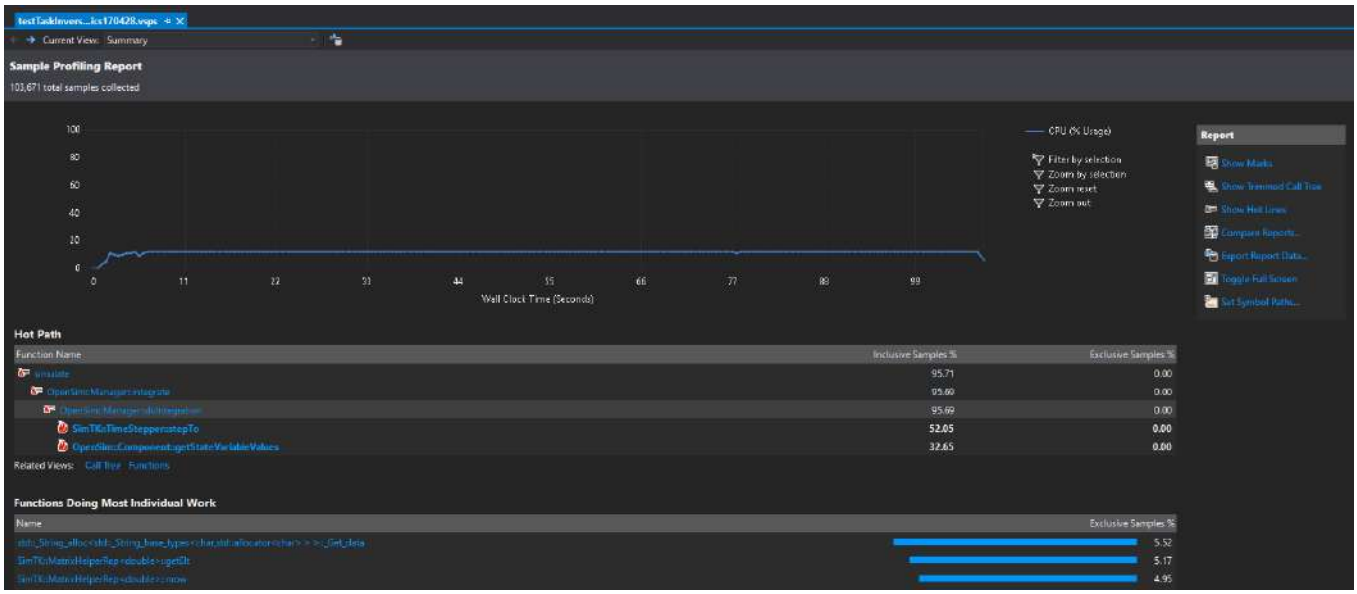


Fig. S-15: Profiling results of TSDIK algorithm. Results show that 95% of the execution time is spent on integrating the system.

G. Evaluation of Motion and Reaction Forces

For evaluating the motion and reaction forces, we will use the previous two body model (Fig. S-11), so that the results can be easily interpreted. We will compare the motion forces of Model-A and Model-B against the generalized forces computed by the ID algorithm (ID). In addition, we will compare the Lagrange multipliers computed from the two models against the joint reaction forces produced by the Joint Reaction Analysis (JRA) (OpenSim). The same sinusoidal movement (Eq. (S-45)) will be used as a benchmark.

Please note that the ID algorithm of OpenSim is not designed to handle constrained dynamics. The reason is that a particular, recorded movement may not satisfy the constraints (e.g. $\phi(q) \neq 0$). In order to compare the validity of the proposed controllers the unconstrained model (Fig. S-11a, with two DoFs) was used in combination with the ID algorithm, while the functionally identical constrained model (Fig. S-11b, with twelve DoFs) was used by the task space controllers (Model-A and Model-B).

Furthermore, a selection matrix has been introduced so that the two models (Fig. S-11) use the same coordinates for actuation. Note that the free body model (Fig. S-11b) contains twelve DoFs that can be used for actuation compared to the two DoFs of the unconstrained model (Fig. S-11a). The active coordinates (actuators) of the model can be selected as follows

$$\tau = B\tau \tag{S-49}$$

where $B \in \Re^{n \times n}$ is a diagonal matrix containing "1" for the coordinates that are used for actuation (active coordinates) and "0" otherwise (passive coordinates). The incorporation of the selection matrix for handling passive coordinates requires further adaptation of the control law that was used in this work. After some derivations (please refer to [35] for more details), it can be shown that in the presence of passive coordinates the control law has the following form

$$\tau = (I - N_{g*}^T[(I-B)N_{g*}^T]^+)(\sum_{i=1}^{g} J_{i|i-1*}^T f_i + N_{g*}^T \tau_0) \tag{S-50}$$

A comparison between the motion forces is shown in Fig. S-16, with a good agreement against the ID algorithm. Please note the restriction imposed by the selection matrix on the passive coordinates. Fig. S-17 presents a comparison of the joint reaction forces, that are applied at the joints as measured in the ground frame, with a good agreement between Model-A Model-B and JRA. We didn't observe similarities between the motion and reaction forces in the case of full actuation ($B = I$), despite that the generated motion is the same.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

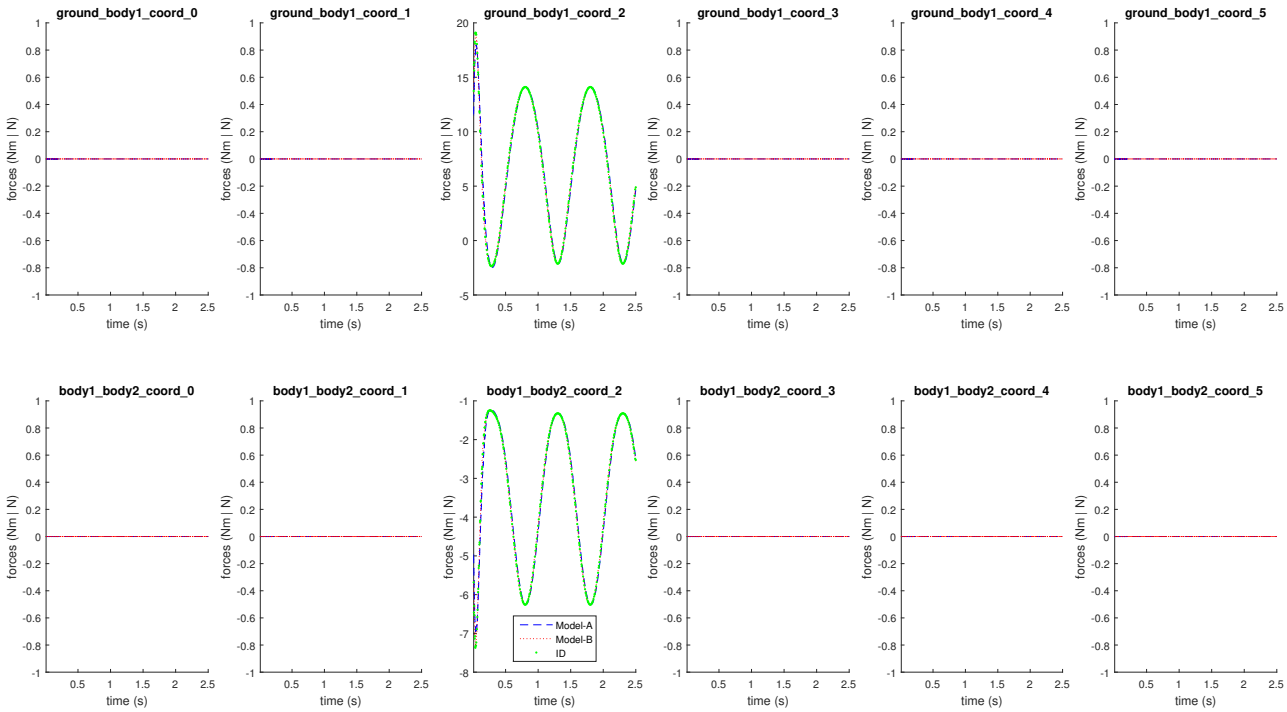


Fig. S-16: Comparison of the motion forces required by Model-A (blue), Model-B (red) and ID (green) for the model presented (Subsection S-D). Each row depicts the 6D spacial forces ($[\tau_x, \tau_y, \tau_z, f_x, f_y, f_z]^T$) applied to a body as required to track the specified motion. The third column compares the torques (z-axis) generated by the two models with the corresponding torques computed by the ID algorithm. Note that ID was applied on the unconstrained model since OpenSim ignores constraint forces in its computation.

A final note on the role of the selection matrix and the differences between Model-A and Model-B. Let's assume that a model contains n unconstrained DoFs, c constraint algebraic equations and p passive DoFs. Then the actual DoFs of the constrained model are of dimension $n_c = n - c$, while the active DoFs are of dimension $n_a = n - p$. We can distinguish three cases [34]:

1) $n_c > n_a$, the system is underactuated. There is at most one solution to the ID problem.
2) $n_c = n_a$, the system is fully actuated. There is a unique solution.
3) $n_c < n_a$, the system is overconstrained. There are an infinite number of solutions (τ) that can achieve the desired movement behavior.

Following this, we conclude that both controllers (Model-A and Model-B) generate identical movement and require the same motion and reaction forces when the system is fully actuated. If the system is overconstrained, then the two controllers will produce different motion and constraint forces, where the former minimizes the inertial weighted command $\tau M^{-1} \tau$, while the latter minimizes the required command $\tau^T \tau$ [34].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBME.2017.2764630, IEEE Transactions on Biomedical Engineering

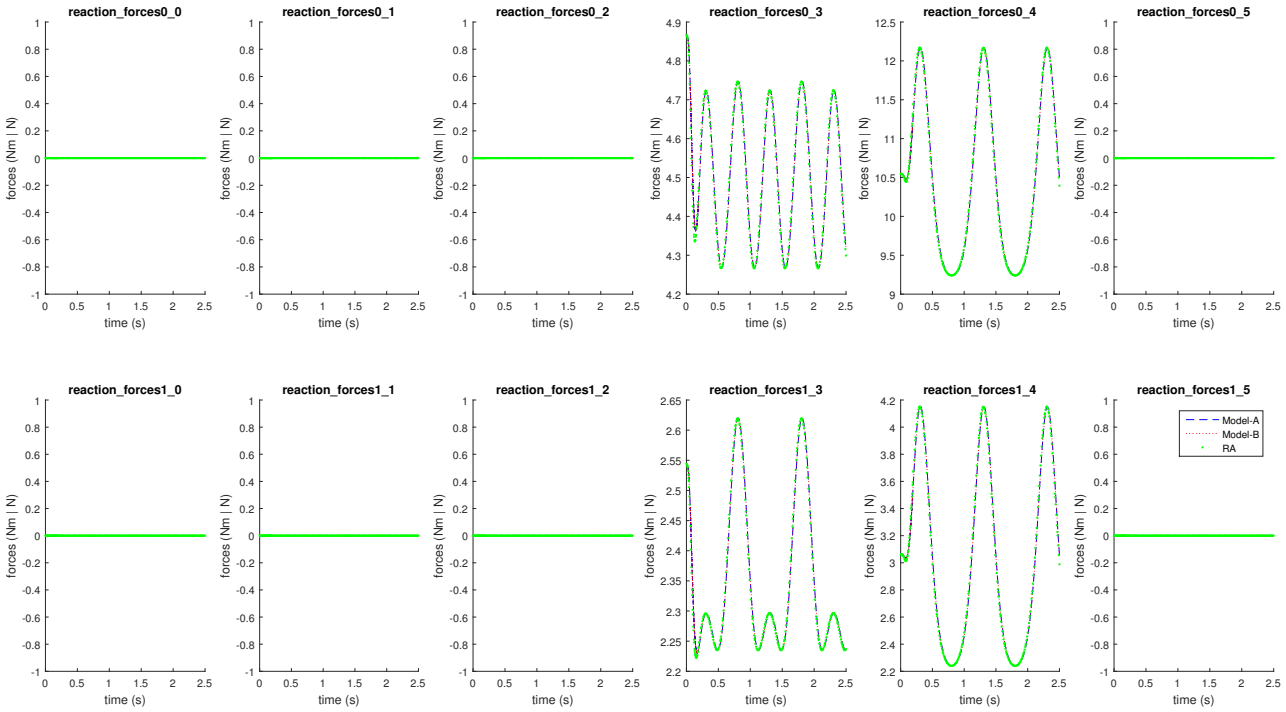


Fig. S-17: Comparison of the joint reaction forces computed by Model-A (blue), Model-B (red) and JRA (green) the model presented (Subsection S-D).