



# Eliminating constraint drift in the numerical simulation of constrained dynamical systems

David J. Braun\*, Michael Goldfarb

Department of Mechanical Engineering, Vanderbilt University, VU Station B, Nashville, TN 37235-1612, USA

## ARTICLE INFO

### Article history:

Received 30 September 2008  
Received in revised form 14 May 2009  
Accepted 21 May 2009  
Available online 3 June 2009

### Keywords:

Constrained dynamics  
Pseudoinverse  
Numerical integration

## ABSTRACT

By means of the Udwadia–Kalaba approach we propose an explicit equation of constrained motion developed to simulate constrained dynamical systems without error accumulation due to constraint drift. The basic idea is to embed a small virtual force and a small virtual impulse to the equation of motion, in order to avoid the drift typically experienced in constrained multibody simulations. The embedded correction terms are selected to minimally alter the dynamics in an acceleration and kinetic energy norm sense. The formulation allows one to use a standard ODE solver, avoiding the need for iterative constraint stabilization. The equation is based on the pseudoinverse of a constraint matrix such that it can be used under redundant constraints and kinematic singularities. The proposed method takes into account the finite word-length of the computational environment, and also accommodates possibly inconsistent initial conditions.

© 2009 Elsevier B.V. All rights reserved.

## 1. Introduction

Constrained dynamical systems are traditionally modeled with a Lagrangian equation of the first kind [1] where additional algebraic variables (Lagrangian multipliers) are used to incorporate the motion constraints to the equation. In addition to this classical approach, many alternative formulations have been proposed in order to model constrained dynamical systems, including: Gauss's principle of least constraint [2], Maggi's equation [3], Gibbs–Appell's formulation [4,5], Kane's equation [6], and the Udwadia–Kalaba approach [7–9], supported with additional discussions and development presented by Pars [10], Neimark and Fufaev [11], Gantmacher [12], Goldstein [13], Chetaev [14] and Lurie [15].

Despite the strong theoretical foundation, direct numerical implementation of the proposed governing equations generally leads to error accumulation due to “constraint drift”. Specifically, motion constraint which should be physically invariant will move in space due to error and imperfection in numerical integration. The resulting solution is not physically consistent and as such loses value with respect to practical interpretation. This issue has been addressed by many authors including: Baumgarte [16], Gear et al. [17], Lötstedt and Petzold [18], Führer and Leimkuhler [19], Petzold [20], ten Dam [21], Eich [22], Bayo and Ledesman [23], Blajer [24] and Aghili [25]. The importance of having a reliable simulation tool which produces physically consistent motion prediction

is motivated by many practical applications as was discussed by Schiehlen [26] and Brogliato et al. [27].

Our aim is to propose a formulation which enables stable numerical simulation without error accumulation in motion constraints. In order to do so, it was necessary to take the nonideal computational environment as well as the possible errors in initial data (caused by the user) into account. Following a revised constraint definition, we derive an explicit equation for constrained motion with constraint correction terms. Although these additional terms have no direct physical meaning, they can be interpreted as a set of small virtual forces and impulses and are derived by means of Gauss's principle of least constraint. After presentation of the proposed formulation, the approach is discussed in the context of prior work in the field. Finally, the approach is illustrated on and validated with several representative examples.

## 2. Constrained multibody dynamics

In this section, the equation of motion for a constrained dynamical system is derived. The approach is based on the explicit equation of constrained motion presented by Udwadia and Kalaba [7].

### 2.1. Unconstrained multibody dynamics

Consider an  $n$ -degree-of-freedom multibody system, the configuration of which is uniquely specified by  $\mathbf{q} \in \mathcal{R}^n$  generalized coordinates. Let the equation of motion of the considered system

\* Corresponding author. Tel.: +1 615 3221760; fax: +1 615 3436925.

E-mail addresses: [david.braun@vanderbilt.edu](mailto:david.braun@vanderbilt.edu) (D.J. Braun), [michael.goldfarb@vanderbilt.edu](mailto:michael.goldfarb@vanderbilt.edu) (M. Goldfarb).

(derived by means of the Lagrangian formalism) be represented in the following form:

$$\mathbf{M}(t, \mathbf{q})\ddot{\mathbf{q}} = \mathbf{Q}(t, \mathbf{q}, \dot{\mathbf{q}}). \quad (1)$$

Here,  $t \in [0, T]$  is a time variable,  $\mathbf{M} \in \mathfrak{R}^{n \times n}$  is a symmetric and positive definite mass matrix while  $\mathbf{Q} \in \mathfrak{R}^n$  represents the generalized forces. If no constraints are applied on (1), the dynamical system is considered as unconstrained with respect to the chosen generalized coordinates  $\mathbf{q}$ .

## 2.2. Holonomic constraints: revised

Let us introduce additional  $m$  holonomic bilateral constraints on the system dynamics,

$$\Phi(t, \mathbf{q}) = \mathbf{0}, \quad (2)$$

where  $\Phi : [0, T] \times \mathfrak{R}^n \rightarrow \mathfrak{R}^m$ . In the forthcoming analysis, we will assume that these rheonomic (explicitly time dependent) constraints are  $C^2[0, T]$ , such that, (2) has well defined first and second partial derivatives at least.

Let us discuss the effect of (2) on the dynamical system (1). Generally speaking, each bilateral constraint adds a constraint reaction force to the system dynamics. If the constraint is ideal, it generates an “ideal reaction” which does no work on any constraint consistent virtual displacement. We assume that all constraints are ideal and as such D’Alembert’s principle applies [28].

Considering the constraint equations (2), one can see that  $\Phi = \mathbf{0}$  defines position-level relations between the generalized coordinates. However, in order to avoid error accumulations along the numerical solution, the holonomic (position) constraints must also be satisfied on the velocity level  $\dot{\Phi} = \mathbf{0}$ . In this light, by adding the aforementioned velocity level constraint to (2), one obtains

$$\Phi(t, \mathbf{q}) = \mathbf{0}, \quad \dot{\Phi}(t, \mathbf{q}, \dot{\mathbf{q}}) = \mathbf{0}. \quad (3)$$

If the dynamical system is restricted by (2), it is important to make sure that not only (2) but also (3) is satisfied. As follows, we will replace (3) with velocity and acceleration level constraints which are linear with respect to  $\dot{\mathbf{q}}$  and  $\ddot{\mathbf{q}}$ , respectively. To do so, let us assume that  $\mathbf{q} = \mathbf{q}(t)$  defines the positions of the constrained dynamical system. Using these functions, one can come up with  $\Phi(t) = \Phi(t, \mathbf{q}(t))$ , and the following Taylor expansion holds

$$\Phi(t + dt) = \Phi(t) + (\mathbf{A}\dot{\mathbf{q}} - \mathbf{b}_q)dt + O(dt^2), \quad (4)$$

where,  $\mathbf{A}(t, \mathbf{q}) = \partial\Phi/\partial\dot{\mathbf{q}}$  and  $\mathbf{b}_q(t, \mathbf{q}) = -\partial\Phi/\partial t$ . Similarly, one can define  $\dot{\Phi}(t) = \dot{\Phi}(t, \mathbf{q}(t), \dot{\mathbf{q}}(t))$  and expand the velocity level (nonholonomic) constraints up to the acceleration level as,

$$\dot{\Phi}(t + dt) = \dot{\Phi}(t) + (\mathbf{A}\ddot{\mathbf{q}} - \mathbf{b}_v)dt + O(dt^2), \quad (5)$$

where  $\mathbf{b}_v(t, \mathbf{q}, \dot{\mathbf{q}}) = -\dot{\mathbf{q}}^T[\partial^2\Phi/\partial\dot{\mathbf{q}}^2]\dot{\mathbf{q}} - 2[\partial^2\Phi/\partial t\partial\dot{\mathbf{q}}]\dot{\mathbf{q}} - \partial^2\Phi/\partial t^2$ .

If  $dt$  is interpreted as a (numerical) integration step, then the first order approximation of (3) (as well as (4) and (5)) can be satisfied at each subsequent integration step,  $\Phi(t + dt) = \mathbf{0}$ ,  $\dot{\Phi}(t + dt) = \mathbf{0}$ , with velocity and acceleration level constraints defined as follows:

$$\begin{aligned} \mathbf{A}\dot{\mathbf{q}} &= \mathbf{b}_q - \Phi/dt, \\ \mathbf{A}\ddot{\mathbf{q}} &= \mathbf{b}_v - \dot{\Phi}/dt. \end{aligned} \quad (6)$$

Instead of (6), the analytical derivation of the explicit equation of constrained motion proposed by Gear et al. [17] and Udwadia and Kalaba [7–9] is based on a velocity or acceleration level representation of the original constraints

$$\mathbf{A}\dot{\mathbf{q}} = \mathbf{b}_q, \quad \mathbf{A}\ddot{\mathbf{q}} = \mathbf{b}_v. \quad (7)$$

However, the equation of constrained motion which is based on (7) does not have a numerically stable ODE implementation without

additional constraint correction. Namely, in order to substitute (6) with (7), the following conditions must be met:

- The user must provide initial conditions which are constraint consistent,  $\Phi(0, \mathbf{q}(0)) = \mathbf{0}$ ,  $\dot{\Phi}(0, \mathbf{q}(0), \dot{\mathbf{q}}(0)) = \mathbf{0}$ .
- The solution  $\mathbf{q}(t)$ ,  $\dot{\mathbf{q}}(t)$  must not contain any integration error.

Due to numerical errors, such conditions are not realistic in a numerical (i.e., computer) simulation environment. Rather, in practice, use of (7) typically leads to significant error accumulation and constraint drift. This motivated us to derive an implementation of (6) which takes the expected numerical imperfections into account. We will show that adequate implementation of (6) prevents error accumulation and results in a numerical solution without constraint drift.

## 2.3. Constrained multibody dynamics

In order to incorporate position-level constraints to the dynamic equations, we recall the Lagrange multiplier approach. However, in contrast to the method traditionally used in mechanics, where the multipliers represent constraint reaction forces, our intention is to use the same idea to eliminate constraint violations.

Let us start with the traditional representation of the constrained dynamical system,

$$\mathbf{M}\ddot{\mathbf{q}} = \mathbf{Q} + \mathbf{Q}_c, \quad (8)$$

where  $\mathbf{Q}_c = \mathbf{A}^T\lambda$  is the generalized constraint force, while the undetermined Lagrange multipliers,  $\lambda \in \mathfrak{R}^m$ , represent the physical forces generated by constraints. This representation is valid for ideal constraints which do no work,  $\mathbf{Q}_c^T\delta\mathbf{q} = 0$ , along any admissible virtual displacement  $\delta\mathbf{q} \in \{\delta\mathbf{q} : \delta\mathbf{q} \in \mathfrak{R}^n, \mathbf{A}\delta\mathbf{q} = \mathbf{0}\}$ . To formulate the equation of motion in explicit form, let us solve the constrained acceleration from (8) as a function of  $\lambda$ ,

$$\ddot{\mathbf{q}} = \mathbf{a} + \mathbf{M}^{-1}\mathbf{A}^T\lambda, \quad (9)$$

where  $\mathbf{a} = \mathbf{M}^{-1}\mathbf{Q}$  is the unconstrained acceleration the system would have without the imposed constraints, see (1). Now, substituting (9) back to the practical acceleration level constraints (6), one can solve for the Lagrangian multipliers by direct inversion

$$\lambda = (\mathbf{A}\mathbf{M}^{-1}\mathbf{A}^T)^{-1}(\mathbf{b}_v - \mathbf{A}\mathbf{a} - \dot{\Phi}/dt). \quad (10)$$

Although, the physical constraint reactions  $\lambda$  are not well defined if the constraint matrix  $\mathbf{A}$  is rank deficient (due to kinematic singularities or constraint redundancy), the constrained acceleration as well as the generalized constraint force  $\mathbf{Q}_c = \mathbf{A}^T\lambda$  are always unique [29,30,8]. Thus, we can determine  $\mathbf{Q}_c$  to accommodate kinematic singularities and constraint redundancy by first defining the following matrices:  $\mathbf{M}^{1/2}$ ,  $\mathbf{M}^{-1/2}$ ,  $\mathbf{B} = \mathbf{A}\mathbf{M}^{-1/2}$  and  $\mathbf{B}^+$ , where due to the positive definiteness of  $\mathbf{M}$ , the so called principal square root of the mass matrix  $\mathbf{M}^{1/2}$  and its inverse  $\mathbf{M}^{-1/2}$  are always well defined, as is the (Moore–Penrose inverse) pseudoinverse  $\mathbf{B}^+$  of  $\mathbf{B}$  [31]. Using the introduced notation, the generalized constraint force becomes

$$\mathbf{Q}_c = \mathbf{M}^{1/2}\mathbf{B}^+(\mathbf{b}_v - \mathbf{A}\mathbf{a} - \dot{\Phi}/dt). \quad (11)$$

Substituting (11) into (8), the explicit equation of the constrained dynamics can be easily obtained. In order to further proceed, one can define the constrained acceleration,  $\ddot{\mathbf{q}} = \dot{\mathbf{v}}$  and rewrite the equation of motion in the following first order form:

$$\begin{aligned} \dot{\mathbf{q}} &= \mathbf{v}, \\ \dot{\mathbf{v}} &= \mathbf{a} + \mathbf{M}^{-1/2}\mathbf{B}^+(\mathbf{b}_v - \mathbf{A}\mathbf{a} - \dot{\Phi}/dt). \end{aligned} \quad (12)$$

Although this formulation accounts for numerical errors on the velocity level,  $\dot{\Phi} \approx \mathbf{0}$ , it cannot in general prevent error accumulation. This is because (12) does not yet take the numerically induced

position-level error given by  $\Phi \approx \mathbf{0}$  into account. With the aim of incorporating this error source, we mimic the above procedure, adding a new Lagrangian multiplier to (12)<sub>1</sub>,

$$\dot{\mathbf{q}} = \mathbf{v} + \mathbf{M}^{-1} \mathbf{A}^T \boldsymbol{\mu}. \quad (13)$$

In contrast to  $\lambda$ , the new multiplier  $\boldsymbol{\mu}$  is not generated by the constraints but rather is introduced to compensate for numerical errors along the integration. A similar term,  $\mathbf{A}^T \boldsymbol{\mu}$  was used by Gear et al. [17] to incorporate velocity level constraints in the equation of motion. Note that neither  $\mathbf{A}^T \boldsymbol{\mu}$  nor  $\mathbf{M}^{-1} \mathbf{A}^T \boldsymbol{\mu}$  has clear physical meaning. However while the former can only be considered as a kinematic correction term, the latter (introduced here) is a dynamic correction which allows us to interpret  $\boldsymbol{\mu}$  as a small mechanical impulse.

Substituting (13) into (6)<sub>1</sub>, one obtains

$$\boldsymbol{\mu} = (\mathbf{A} \mathbf{M}^{-1} \mathbf{A}^T)^{-1} (\mathbf{b}_q - \mathbf{A} \mathbf{v} - \Phi/dt). \quad (14)$$

Once again, using the pseudoinverse notation, the compensation term becomes

$$\mathbf{M}^{-1} \mathbf{A}^T \boldsymbol{\mu} = \mathbf{M}^{-1/2} \mathbf{B}^+ (\mathbf{b}_q - \mathbf{A} \mathbf{v} - \Phi/dt). \quad (15)$$

Substituting (15) into (13), and combining with (12)<sub>2</sub>, the complete equation of motion for the constrained dynamical system is obtained

$$\begin{aligned} \dot{\mathbf{q}} &= \mathbf{v} + \mathbf{M}^{-1/2} \mathbf{B}^+ (\mathbf{b}_q - \mathbf{A} \mathbf{v} - \Phi/dt), \\ \dot{\mathbf{v}} &= \mathbf{a} + \mathbf{M}^{-1/2} \mathbf{B}^+ (\mathbf{b}_v - \mathbf{A} \mathbf{a} - \dot{\Phi}/dt), \end{aligned} \quad (16)$$

where  $\mathbf{a} = \mathbf{M}^{-1} \mathbf{Q}$  is the unconstrained acceleration,  $\dot{\mathbf{v}}$  is the constrained acceleration, and  $\dot{\mathbf{q}}$  is the constrained velocity.

It can be easily recognized that if  $\Phi = \mathbf{0}$  and  $\dot{\Phi} = \mathbf{0}$  (which also implies  $\mathbf{A} \mathbf{v} = \mathbf{b}_q$ ) then (16) reduces to the well known explicit equation of motion derived by Udwadia and Kalaba (which assumes an ideal computational environment and perfect initial conditions). Taking the error sources in the real computational environment into account, we do not assume exact constraint satisfaction which, following strict mathematical derivations, produces additional error compensation terms in the equation. The new terms compensate for the numerical errors and guarantee that no error accumulation can take place.

#### 2.4. Discussion of the proposed formulation

In order to discuss (16), let us recall the explicit equation of motion which does not contain the constraint correction terms,

$$\begin{aligned} \dot{\mathbf{q}} &= \mathbf{v}, \\ \dot{\mathbf{v}}^i &= \mathbf{a} + \mathbf{M}^{-1/2} \mathbf{B}^+ (\mathbf{b}_v - \mathbf{A} \mathbf{a}), \end{aligned} \quad (17)$$

where  $\dot{\mathbf{v}}^i = \mathbf{a} + \mathbf{R}^{-1} \mathbf{C}_v^+ (\mathbf{b}_v - \mathbf{A}_v \mathbf{a})$  is the acceleration which exactly satisfies  $\mathbf{A}_v \dot{\mathbf{v}}^i = \mathbf{b}_v$ , while  $\mathbf{v}$  is the velocity obtained by numerical integration of  $\dot{\mathbf{v}}^i$  (in an ideal computational environment one would obtain,  $\mathbf{v} = \mathbf{v}^i$ ).

Comparing (16)<sub>2</sub> with (17)<sub>2</sub>, one may conclude that  $\mathbf{M}^{-1/2} \mathbf{B}^+ \dot{\Phi}/dt$  represents a small correction of the constraint force (Lagrangian multiplier  $\lambda$ , see (10)) which is necessary to satisfy the constraints under numerically imperfect conditions. In order to have a clearer interpretation, let us reformulate (16)<sub>2</sub> by means of the Gauss principle of least constraint,

$$\dot{\mathbf{v}} = \min\{\mathbf{x} \in \mathfrak{R}^n : (\mathbf{x} - \dot{\mathbf{v}}^i)^T \mathbf{M} (\mathbf{x} - \dot{\mathbf{v}}^i), \mathbf{A} \mathbf{x} = \mathbf{b}_v - \dot{\Phi}/dt\}. \quad (18)$$

Based on this interpretation, the  $\dot{\mathbf{v}}$  provided by (16)<sub>2</sub> is the closest acceleration to  $\dot{\mathbf{v}}^i$  (in an acceleration energy sense) which satisfies the constraints (6)<sub>2</sub>.

Similarly, comparing the first equations in (16) and (17), one might recognize that  $\mathbf{M}^{-1/2} \mathbf{B}^+ (\mathbf{b}_q - \mathbf{A} \mathbf{v} - \Phi/dt)$ , although not generated by physical constraints, is necessary to prevent error accu-

mulation. In order to give a physical interpretation of this term, let us define an equivalent formulation of (16)<sub>1</sub> with the following constrained quadratic program

$$\dot{\mathbf{q}} = \min\{\mathbf{x} \in \mathfrak{R}^n : (\mathbf{x} - \mathbf{v})^T \mathbf{M} (\mathbf{x} - \mathbf{v}), \mathbf{A} \mathbf{x} = \mathbf{b}_q - \Phi/dt\}. \quad (19)$$

Here,  $\dot{\mathbf{q}}$  is the closest velocity to  $\dot{\mathbf{v}}$  (in a kinetic energy sense) which satisfies the constraints (6)<sub>1</sub>. Note that  $\mathbf{v}$ , obtained by time integration of the constrained acceleration, may not satisfy exactly the kinematic constraints  $\mathbf{A} \mathbf{v} \neq \mathbf{b}_q$  (which condition is taken into account along the derivation). In a special case when  $\mathbf{v} = \mathbf{v}^i$ , ( $\mathbf{A} \mathbf{v} = \mathbf{b}_q$ ), the correction term would reduce to  $-\mathbf{M}^{-1/2} \mathbf{B}^+ \Phi/dt$ .

One can conclude now that the derivation of the acceleration level correction terms are closely related to the general principle of constrained motion formulated by Gauss [2], while the velocity level correction terms are obtained using kinetic energy minimization, and as such are also physically motivated.

### 3. Numerical implementation

In order to simulate a constrained dynamical system, a robust and stable numerical solver for the index-3 differential-algebraic equation (DAE) (1) and (2), is required [32]. However, in contrast to widely available ordinary differential equation (ODE) solvers, a DAE solver which reliably prevents error accumulation and constraint drift is not trivial to implement. By means of DAE integration, DASSL [33] (and its extended version DASSLRT) offers a state-of-the-art implementation of the index-2 DAE formulation proposed by Gear et al. [17]. In general, different DAE integrators have been developed as research codes, overviews of which can be found in [34–36].

Our intention is to show that one can use traditional ODE integrators to solve the reformulated constrained dynamic equation (16), without having problems of error accumulation and constraint drift. In the remainder of this section, we discuss how to obtain such a numerical solution.

When the analytical model is derived, the system is characterized with the following quantities;  $\mathbf{M}$ ,  $\mathbf{h}$ ,  $\mathbf{Q}$ ,  $\mathbf{A}$ ,  $\mathbf{b}_q$ ,  $\mathbf{b}_v$ ,  $\Phi$  and  $\dot{\Phi}$ . Without any further preparation, (16) is ready to be solved in a standard ODE solver which utilizes a first order state-space formulation, providing we can incorporate the correction terms,  $\Phi/dt$  and  $\dot{\Phi}/dt$  where, as it was mentioned,  $dt$  is interpreted as a time step of the numerical integrator. The simplest way to incorporate the correction terms is to use a fixed step solver where  $dt$  is predefined. If however, one wants to exploit the benefits of a variable step solver, the actual time step should be used over the integration procedure.

The computational expense of the numerical implementation of (16) is dominated by the calculations of the principal square root of the mass matrix  $\mathbf{M}^{1/2}$  and the pseudoinverse  $\mathbf{B}^+$ . Practically, these computations entail an eigenvalue computation of  $\mathbf{M}$  and a singular value decomposition for  $\mathbf{B}$ , see [37]. However, in order to incorporate the constraint forces, one can use  $\mathbf{M}^{-1/2} \mathbf{B}^+ = \mathbf{R}^{-1} \mathbf{C}^+$  where  $\mathbf{R}$  is the upper triangular Cholesky factor of the inertia matrix  $\mathbf{M} = \mathbf{R}^T \mathbf{R}$  while  $\mathbf{C} = \mathbf{A} \mathbf{R}^{-1}$ . This replacement allows one to avoid the particularly expensive eigenvalue computation which would be required in order to compute  $\mathbf{M}^{1/2}$ . It is also important to mention that the pseudoinverse notation utilized in (16) allows a compact and general representation of the equation of motion regardless of whether the constraints are independent or dependent. From a computational point of view, however, only dependent constraints require singular value decomposition to define  $\mathbf{B}^+$  (or  $\mathbf{C}^+$ ) while for independent constraints one can either compute the velocities and accelerations together with the Lagrangian multipliers from (9), (10), (13) and (14), or utilize

the following explicit definition,  $\mathbf{M}^{-1/2}\mathbf{B}^+ = \mathbf{R}^{-1}\mathbf{C}^T(\mathbf{C}\mathbf{C}^T)^{-1}$  to evaluate the right hand side of (16) directly.

Note, that the correction terms are obtained based on the Taylor expansion of the constraints, see (4) and (5), which provides a good approximation as long as a small integration step  $dt$  is used. With this in mind, one cannot expect arbitrarily precise constraint satisfaction (i.e., due to the finite time step and numerical imprecision). Let us recognize that the correction terms in (16) use the same matrix  $\mathbf{M}^{-1/2}\mathbf{B}^+$ , which must in any case be computed in order to incorporate the constraint forces. In this light, beyond multiplication and addition, the correction terms do not require any additional computation.

In the following, a simple numerical implementation of (16) will be given with some practical comments.

### 3.1. Numerical procedure

Using a small time step  $dt$ , the finite domain of integration  $t \in [0, T]$  is equidistantly discretized as  $0 = t_0 < t_1 < \dots < t_n < t_{n+1} < \dots < t_N = T$ . By means of numerical solution, we seek the discrete values of all positions  $\{\mathbf{q}_0, \mathbf{q}_1, \dots, \mathbf{q}_n, \mathbf{q}_{n+1}, \dots, \mathbf{q}_N\}$  and velocities  $\{\mathbf{v}_0, \mathbf{v}_1, \dots, \mathbf{v}_n, \mathbf{v}_{n+1}, \dots, \mathbf{v}_N\}$  which are numerically constraint consistent. Let us assume that the initial conditions  $\mathbf{q}_0 = \mathbf{q}(0)$  and  $\mathbf{v}_0 = \mathbf{v}(0)$  approximately satisfy the constraints,  $\Phi(t_0, \mathbf{q}_0) \approx \mathbf{0}$ ,  $\dot{\Phi}(t_0, \mathbf{q}_0, \mathbf{v}_0) \approx \mathbf{0}$ . The complete solution can thus be obtained by integrating the system dynamics successively between discrete time instants. In order to illustrate salient aspects of the implementation, a numerical integration of (16) is presented based on a forward Euler method, as follows:

- (1) Based on  $\mathbf{q}_n = \mathbf{q}(t_n)$  and  $\mathbf{v}_n = \mathbf{v}(t_n)$ , known from the previous integration step (or defined by initial conditions for the starting step), one can evaluate:  $\mathbf{Q} = \mathbf{Q}(t_n, \mathbf{q}_n, \mathbf{v}_n)$ ,  $\mathbf{M} = \mathbf{M}(t_n, \mathbf{q}_n)$ ,  $\mathbf{A} = \mathbf{A}(t_n, \mathbf{q}_n)$ ,  $\mathbf{b}_q = \mathbf{b}_q(t_n, \mathbf{q}_n)$ ,  $\mathbf{b}_v = \mathbf{b}_v(t_n, \mathbf{q}_n, \mathbf{v}_n)$ ,  $\Phi = \Phi(t_n, \mathbf{q}_n)$  and  $\dot{\Phi} = \dot{\Phi}(t_n, \mathbf{q}_n, \mathbf{v}_n)$ . The upper triangular Cholesky factor of the mass matrix  $\mathbf{R}$  is computed, where  $\mathbf{M} = \mathbf{R}^T\mathbf{R}$ , and the pseudoinverse  $\mathbf{C}^+$  is computed based on  $\mathbf{C} = \mathbf{A}\mathbf{R}^{-1}$ .
- (2) Using  $\mathbf{Q}$ ,  $\mathbf{h}$  and exploiting the Cholesky factorization one can efficiently solve the unconstrained acceleration  $\mathbf{a}_n$  from  $\mathbf{R}^T\mathbf{R}\mathbf{a}_n = \mathbf{Q}$  with a successive forward and backward substitution.
- (3) The endpoint position is computed from:

$$\mathbf{q}_{n+1} = \mathbf{q}_n + \mathbf{v}_n dt + \mathbf{R}^{-1}\mathbf{C}^+[(\mathbf{b}_q - \mathbf{A}\mathbf{v}_n)dt - \Phi], \quad (20)$$

and the endpoint velocity from:

$$\mathbf{v}_{n+1} = \mathbf{v}_n + \mathbf{a}_n dt + \mathbf{R}^{-1}\mathbf{C}^+[(\mathbf{b}_v - \mathbf{A}\mathbf{a}_n)dt - \dot{\Phi}]. \quad (21)$$

At the end of an integration step, the new position and velocity ( $\mathbf{q}_{n+1}, \mathbf{v}_{n+1}$ ) is obtained. These values are used to initialize the next integration cycle.

The presented method assures that neither use of initial conditions which do not exactly satisfy the constraints, the roundoff error (caused by imperfect arithmetics), nor the truncation error (made by discretization) can cause constraint drift along the time integration. This however, does not mean that inconsistent initialization is preferred. Namely, poorly selected  $\mathbf{q}_0$  and  $\mathbf{v}_0$  will cause intensive corrections at the beginning of the integration which may alter the dynamic evolution of the system over time. To avoid this effect, it is important to take care of correct initialization using approximately (numerically) consistent initial conditions,  $\Phi(t_0, \mathbf{q}_0) \approx \mathbf{0}$ ,  $\dot{\Phi}(t_0, \mathbf{q}_0, \mathbf{v}_0) \approx \mathbf{0}$  [38,39].

Let us mention that due to  $1/dt$  in (6), the compensation terms may make the governing equation (16) stiff. However, this

term is canceled after time discretization as it is shown in (20) and (21).

If the constraints are independent, the pseudoinverse used in the numerical procedure is explicitly defined  $\mathbf{C}^+ = \mathbf{C}^T(\mathbf{C}\mathbf{C}^T)^{-1}$ . In general however,  $\mathbf{C}^+$  requires a singular value decomposition of  $\mathbf{C}$ . This computation may be relatively expensive, although it allows one to resolve kinematic singularities and generally handle dynamical systems with dependent constraints. For high degree-of-freedom multibody systems however, reducing the computational effort needed for time integration could become crucial. In this case, one may favor Cartesian or “natural” coordinates [40,35,41], and use topological based approaches to exploit the structure as well as the sparsity pattern of the formulation [42,43].

In order to improve accuracy and/or numerical stability, instead of the presented simple scheme, more sophisticated explicit or implicit discretization can be used as required. Implicit integrators are computationally more expensive, but also more stable and are usually required if the equations of motion are stiff. Practically, an implicit solver would use Newton iteration to obtain the positions and velocities at each time step. This iterative process can be sped up by exploiting the sparsity pattern of the Jacobian used in Newton’s method as is proposed in [44]. Note however that due to its computational expense, solving (16) with an implicit solver is only reasonable if the constraints are independent, in which case no singular value decomposition is required to compute  $\mathbf{C}^+$ .

## 4. Related simulation methods

In this section, established DAE integration methods frequently used to prevent constraint drift are discussed with respect to the presented equation (16). An overall review of constraint enforcement approaches can be found in [45,46].

### 4.1. Baumgarte’s constraint stabilization

The main issue in all DAE problems is ensuring that the small numerical error made at each integration step does not accumulate along the solution process. One of the most popular methods in engineering practice, which does not require iterative constraint corrections, is Baumgarte’s constraint stabilization [16]. Practically, instead of using an original constraint  $\Phi = 0$ , Baumgarte proposes the use of a corresponding second order equation,  $\ddot{\Phi} + 2\alpha\dot{\Phi} + \beta^2\Phi = 0$  which for  $\alpha = \beta > 0$  has a globally asymptotically stable aperiodic solution approaching zero ( $\Phi=0$ ) over time. As has been frequently pointed out in the literature, the introduced parameters,  $\alpha$  and  $\beta$  must be carefully selected, since the selection can make the reformulated problem stiff and also can alter the original dynamics of the system under consideration. In (16), the correction terms are derived to minimally alter the dynamics under numerically imperfect conditions without resulting in stiff equations. This was achieved by means of (18) and (19), without introducing extraneous parameters like  $\alpha$  and  $\beta$ .

Various modifications of Baumgarte’s idea can be found in [47–50]. Specifically, the approach proposed by Ascher et al. [49] is based on a single correction step (Newton iteration step) towards the position and velocity constraint manifold (3) applied after each time step. In the present paper, the velocity and acceleration level constraint set (6) can also be seen as a Newton iteration algorithm for (3). However, in the present approach, the correction terms embedded in (16), are derived to minimally alter the uncorrected solution in a kinetic and acceleration energy sense according to (18) and (19), and as such they cannot be obtained by directly solving (6) as proposed in [49]. Moreover, instead of post-correcting the computed solution, one can recognize in (20) and (21) that

with explicit discretization, the correction terms derived here perform pre-correction.

#### 4.2. Iterative correction approach

In contrast to Baumgarte's approach, a variety of other methods have been developed which eliminate constraint drift by iteratively correcting the already computed solution.

The penalty based Augmented Lagrangian (AL) formulation introduced by Bayo and Ledesman [23] works with redundant constraints and singular mass matrices. In order to deal with constraint violation, a "mass-orthogonal" projection method was formulated on the acceleration, velocity and position level. Based on results presented in [23], the method can provide numerically perfect constraint satisfaction along relatively large step integrations. The motivation of the mass-orthogonal algorithm is to have the same matrix for the dynamic equations and also for the iterative constraint correction process.

The correction approach proposed in this paper can also be seen as mass-orthogonal. However, the idea presented herein is based on Gauss's principle, and as such it is free of the auxiliary (penalty) parameter one needs to specify on the AL formulation.

An alternative two-step decoupled position and velocity level constraint correction algorithm had also been proposed by Blajer [24]. This approach is built on the geometric interpretation of the constrained motion [51]. The method assumes a full-rank constraint matrix and as such it cannot be used for simulation of dynamical systems with redundant constraints.

Aghili [25] presented an efficient formulation for the constraint motion problem introducing a "constraint inertia matrix". In order to satisfy the (possibly dependent) position constraints, he proposed a geometrically motivated correction method based on the pseudoinverse of the constraint matrix.

The approach presented in this paper does not use the pseudoinverse of the original constraint matrix, but rather is based on the inertially weighted pseudoinverse, which allows dynamically consistent constraint correction (i.e., based on Gauss's principle).

#### 4.3. Coordinate partitioning method

In many practical applications, using dependent coordinates  $\mathbf{q}$  with differential-algebraic equations (1) and (2) is a convenient and natural way to model constrained dynamical systems. However, having constraints, and as such dependent coordinates, is the primary reason for the constraint drift along a numerical integration. Theoretically, this problem can be overcome by analytical reformulation of the mathematical model, embedding all constraints in (1) by specially selected (independent) generalized coordinates. This kind of reformulation of the original DAE problem to a corresponding ODE is often nontrivial or even impossible in practice. In this light, Wehage and Haug [52] proposed a more practical coordinate partitioning in order to separate the dependent coordinates from the independent ones. This partitioning, although nontrivial and not unique, exactly eliminates the drift at the velocity level from the integration, and allows DAE problems to be solved accurately using a correction only on the position level.

#### 4.4. Differential-algebraic approach

In order to incorporate the motion constraints to the governing equation, Gear [32] developed an index reduction method. Instead of solving the original index-3 DAE problem (1) and (2), he proposed an alternative index-2 DAE formulation [17], where the velocity level constraints,  $\mathbf{A}\dot{\mathbf{q}} = \mathbf{b}_q$  were directly embedded in the equation of motion. In contrast to the Udwadia–Kalaba approach, Gear did not eliminate the Lagrangian multipliers but rather calcu-

lated these in each time step. The integration was based on the Backward Differentiation Formula combined with Newton iterative correction of the original constraints  $\Phi$ . Similar methods have also been developed by Lötstedt and Petzold [18], Führer and Leimkuhler [19] and Eich [22].

In the presented approach, the Lagrangian multipliers are not computed, which allows us to take redundant constraints into account. On the other hand, instead of using the velocity (and acceleration) level constraints in standard form, we have used (6), which allows (16) to prevent error accumulation without iteration.

Note that Eich [22] has proposed a quadratic minimization based method which corrects the computed solution by projecting it to the position and velocity constraint (3) after each time step. The difference between the approach suggested in [22] and the method presented herein is twofold; first, the minimization we propose is performed in an acceleration and kinetic energy sense, and second, the velocity and acceleration level constraints utilized in this paper allows the correction approach to be explicitly incorporated in Eq. (16).

#### 4.5. Taking the numerical error sources into account

The importance considering the finite word-length of the computational platform, as well as the inconsistency of the initial data was also pointed out by ten Dam [21]. In contrast to Gear [32], ten Dam argued that the index of the DAE is not what causes difficulties in the solution, but rather the order of steps one takes to obtain the discrete formulation. Instead of discretizing the analytically derived equations, ten Dam proposed deriving the discrete Lagrange multipliers with the primary objective of forcing the solution to satisfy the constraints at each time step. It was shown that the discrete multipliers are not equal to the discretized version of the analytically derived multipliers, which is considered the main reason for the numerical instability experienced by standard approaches.

From our viewpoint, the primary reason of the error accumulation and constraint drift lies in the standard constraint representation (7). Namely, it assumes  $\Phi = \mathbf{0}$ ,  $\dot{\Phi} = \mathbf{0}$  along the numerical solution, and as such eliminates the information from the drift. Under this assumption, correction of the constraint drift is not possible. Taking the finite word-length of the computational environment into account, we accept  $\Phi \approx \mathbf{0}$ ,  $\dot{\Phi} \approx \mathbf{0}$ , and use the velocity and acceleration level constraints independently as is proposed with (6). In this way the constraint drift can naturally and automatically be eliminated.

#### 4.6. On the presented method

In order to obtain unique Lagrange multipliers, it is traditionally assumed that the constraint matrix  $\mathbf{A}$  is of full rank. In real simulation however, one cannot guarantee this property through the dynamical evolution of the constrained motion. Particularly, if the constraint set becomes dependent,  $\mathbf{A}$  loses rank, and the Lagrange multipliers cannot be uniquely calculated, and as such the numerical simulation fails. By this reason, the approach presented here does not require the Lagrange multipliers to be computed, instead, it uses the generalized constraint force which is always well defined.

All the dynamical simulations which seek precise constraint satisfaction implement some type of correction algorithm. This correction process alters the dynamic evolution of the system and can make it depart from the expected natural behavior over time. In order to minimize this effect, the proposed correction terms are derived to minimally alter the motion in an acceleration and kinetic energy norm sense.

In contrast to the frequently used iterative type constraint corrections, the equation of motion (16) proposed here, does not re-

quire any iteration, which may be highly preferred in real time applications. The noniterative constraint correction is achieved by directly embedding (6) in the equation of motion. Practically, this allows us to obtain one corrective step (toward the position and the velocity level constraints) at each time step. Note, however, that because only one corrective step is allowed, using initial conditions which significantly violate the constraints will result in a low accuracy solution. In this light, it is important to use initial conditions which at least approximately satisfy the constraints and thus can be accepted as constraint consistent in a numerical sense.

Finally, it is important to mention that utilizing the idea presented in [9], (16) can also be generalized to accommodate nonholonomic and nonideal constraints.

### 5. Application

In this section, the proposed formulation (16) is first tested on relatively simple dynamical systems, then on a more complex one. Specifically, in order to assess the accuracy of the solution, examples are chosen which are sufficiently simple to allow formulation of the dynamics without explicitly imposed constraints, which in turn enables an ODE formulation of the dynamics. The ODE formulation and subsequent solution via an ODE solver, which is termed the “trusted solution”, is used in these cases to compare the results of the respective DAE solution (16) termed as the “proposed solution” (both under the same numerical conditions). The authors also compare the DAE solutions obtained by (16) and (17) (i.e., with and without constraint error correction) for constraint drift in the presence of “perfectly” consistent and inconsistent initialization. After comparing solutions for the simple examples, a seven-link biped (in three different constraint configurations), which is too complex to be formulated as a single ODE problem, is used as a “realistic” application of the proposed method.

The presented examples are solved with a fourth order fixed step Runge–Kutta method. The solver is implemented in MATLAB and compiled to a C code. Using this code, the simulations are performed on 2.4 MHz Intel Core2 Quad PC computer with a fixed time step. The selected integration step preserves stability of the explicit Runge–Kutta integrator. The error of the reported numerical solutions are measured with respect to the “numerically exact solution” obtained using the ODE formulation integrated with a MATLAB solver (with  $10^{-12}$  relative and absolute tolerance). All physical quantities used in the simulations have standard SI units [kg,m,s].

#### 5.1. Mathematical pendulum

A pendulum with mass  $m$  and length  $l$  is chosen to test the proposed method over a long time period simulation  $t \in [0, 1000]$  s. The constrained equation of motion, (16), is derived using two (dependent) coordinates  $\mathbf{q} = [x, y]^T$  and one constraint  $\Phi = x^2 + y^2 - l^2$ . Starting from the horizontal rest position, the motion is simulated using  $10^{-3}$  time step. Compared to the “numerically exact” position of the pendulum, the simulation results show the same accuracy,  $3 \times 10^{-7}$  (i.e., worst error magnitude), for the “proposed solution” and the “trusted solution”, indicating that this error is not due to the constrained formulation but is rather generated by discretization. On the other hand, using (17) (with no constraint correction) results in a low accuracy solution,  $10^{-1}$ , as expected. In order to identify the importance of the correction terms, the simulations were repeated with slightly imperfect initial conditions. The result is depicted in Fig. 1. Note that despite the imperfect initialization, no error accumulation took place in the

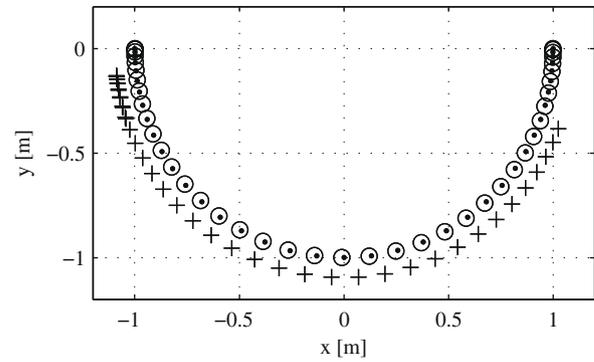


Fig. 1. Mathematical pendulum:  $l = 1$  m,  $m = 1$  kg. Last swing in  $t \in [0, 1000]$  s simulation is depicted. The “trusted solution” started from the rest horizontal position  $\mathbf{q}(0) = [1, 0]^T$ ,  $\mathbf{v}(0) = [0, 0]^T$ , is plotted with “+”. The solution obtained by (16), with imperfect initial conditions  $\mathbf{q}(0) = [1 + 10^{-5}, 10^{-5}]^T$ ,  $\mathbf{v}(0) = [10^{-4}, -10^{-4}]^T$ , is depicted with “o”. The solution obtained by (17) with the same imperfect initialization is plotted with “+”. This solution violates the constraint and is shifted in time.

proposed solution. Correspondingly (except the first few steps) the constraint is satisfied up to,  $10^{-12}$ , such that the solution is practically free of drift. Let us mention that integration of (16) over 1000 s took 13.7 s CPU time, 26% of which was spent on constraint correction.

Since in the present context an explicit solver is utilized, the overall accuracy of the integrated solution (accuracy compared to the numerically exact solution) as well as the accuracy of the constraint satisfaction is step-size dependent. A representative rela-

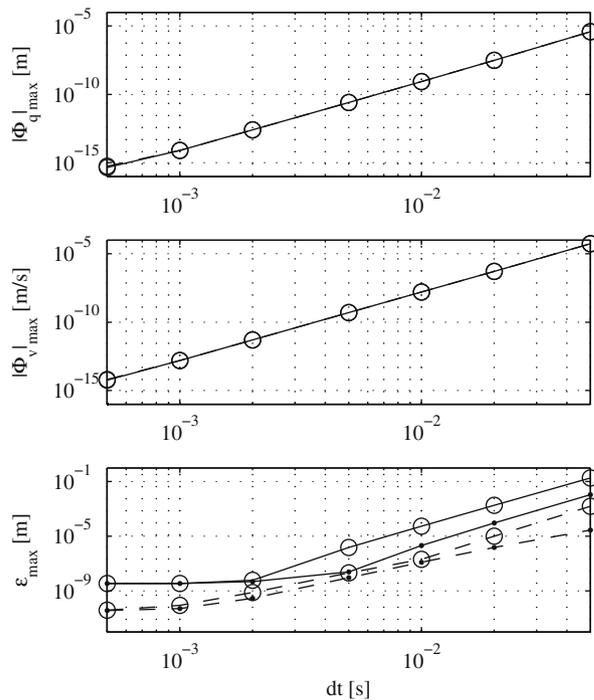


Fig. 2. In this figure, log–log pictures depict the maximum constraint error  $|\Phi_q|_{\max}$ ,  $|\Phi_v|_{\max}$ , and the maximum overall accuracy  $\epsilon_{\max}$ ,  $\epsilon = \sqrt{dx^2 + dy^2}$  (where  $dx$  and  $dy$  are errors in the corresponding coordinates) between the “trusted solution”, “+”, and the proposed solution, “o”. The presented results are based on integration conducted for  $t \in [0, 10]$  s (dashed line) and  $t \in [0, 100]$  s (full line) with seven different time steps:  $dt \in [5 \times 10^{-4}, 10^{-3}, 2 \times 10^{-3}, 5 \times 10^{-3}, 10^{-2}, 2 \times 10^{-2}, 5 \times 10^{-2}]$ . In the first two figures, the full and the dashed lines are overlapped, indicating that the two solutions have the same accuracy on the constraint satisfaction.

tion between the integration step-size and the mentioned accuracy measures is illustrated in Fig. 2. As one can recognize, with a small enough time step, the overall accuracy obtained with the proposed formulation is the same as that obtained by integration of the unconstrained formulation. The presented numerical result also verifies that with a larger time step, the constrained formulation gives a less precise result, namely the constraint error (although steadily maintained) is not in the order of the machine precision.

5.2. Slider-crank mechanism

Consider the slider-crank mechanism consisting of two links with equal lengths  $l$  and masses  $m$ , and a horizontal slider, with mass  $m_s$ , attached to the end of the mechanism (Fig. 3). When the links are either horizontal or vertical the mechanism is in a singular configuration. Our intention is to test (16) under this singularity. To make the motion periodically cross the singular positions, the base link is subjected to a constant counterclockwise

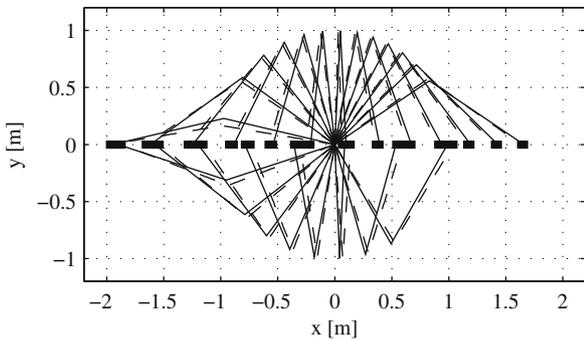


Fig. 3. Slider-crank mechanism:  $l = 1$  m,  $m = 1$  kg,  $m_s = 1$  kg. Stroboscopic view of the motion for  $t \in [97.5, 99]$  s. The solution obtained by (16) with exact initialization  $\mathbf{q}(0) = [0, 0, 0, 0]^T$ ,  $\mathbf{v}(0) = [0, 0, 0, 0]^T$ , is depicted with “-”. The numerical integration took 18.9 s CPU time, 5% of which is spent on constraint correction. The motion predicted by (16) under slightly inconsistent initialization  $\mathbf{q}(0) = [10^{-3}, 10^{-2}, 10^{-3}, 10^{-2}]^T$ ,  $\mathbf{v}(0) = [10^{-3}, 10^{-2}, 10^{-3}, 10^{-2}]^T$  is plotted with “- -”. The solution obtained by (17) (with no constraint correction), was highly inaccurate such that we decided not to present it here.

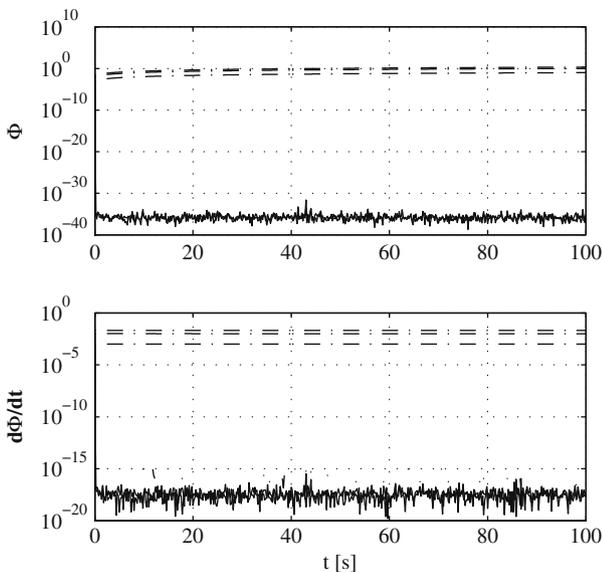


Fig. 4. Constraint evolution along the solution obtained by (16) is plotted with “-”. Note that all constraints are below the machine precision  $10^{-15}$ . Constraint evolution along the solution obtained with no constraint correction is depicted with “- -”. Both depicted solutions started with inconsistent initialization.

torque of 20 N m and a torsional (linear) viscous damping with damping coefficient 5 N m s. The constrained formulation is derived with four (dependent) coordinates  $\mathbf{q} = [x_1, y_1, \theta_1, \theta_2]^T$  and three constraints  $\Phi = [x_1, y_1, y_1 + l(\sin(\theta_1) + \sin(\theta_2))]^T$ . Here,  $(x_1, y_1)$  are coordinates of the support point while  $\theta_1$  and  $\theta_2$  are absolute angles of the links measured counterclockwise from a horizontal reference.

The motion, started from a horizontal rest position, is simulated over  $t \in [0, 100]$  s with  $10^{-3}$  time step. Compared to the numerically exact solution, the motion reflected to the horizontal position of the slider  $x_s = x_1 + l(\cos(\theta_1) + \cos(\theta_2))$ , shows the same accuracy,  $4.4 \times 10^{-8}$ , for the proposed solution and the trusted solution. Eq. (16) is also tested under inconsistent initialization. The simulation results are depicted in Figs. 3 and 4.

5.3. Two four-bar linkages

Let us consider two four-bar linkages with links of length  $l$  and distributed masses  $m$ . When the mechanism moves through a horizontal position, its number of degrees of freedom changes

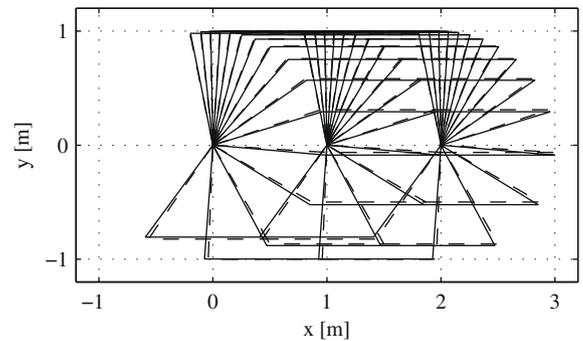


Fig. 5. Two four-bar linkages:  $l = 1$  m,  $m = 1$  kg for each link. Stroboscopic view of the motion for  $t \in [95, 96.25]$  s obtained by (16) with exact initialization  $\mathbf{q}(0) = [0, 1, 1, 2, 1]^T$ ,  $\mathbf{v}(0) = [1, 0, 1, 0, 1, 0]^T$ , is depicted with “- -”. The integration took 5.2 s CPU time. The motion predicted by (16) under slightly inconsistent initialization  $\mathbf{q}(0) = [0, 1 + 10^{-2}, 1, 1 + 10^{-2}, 2 - 10^{-2}, 1]^T$ ,  $\mathbf{v}(0) = [1 - 10^{-2}, 0, 1, 0, 1 + 10^{-2}, 0]^T$  is plotted with “-”.

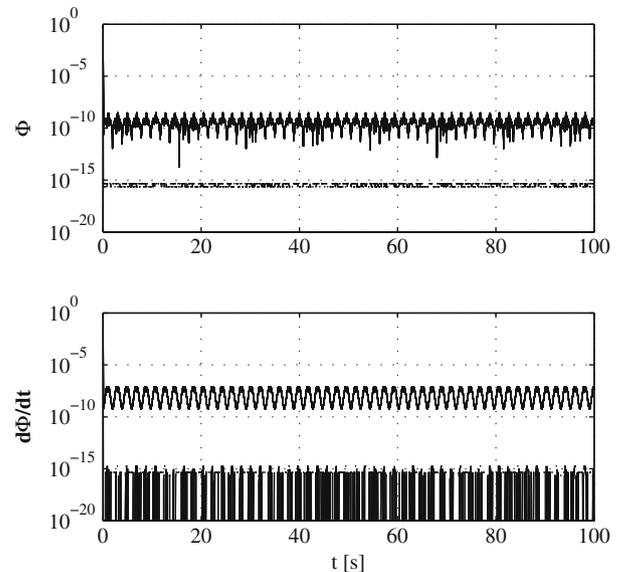


Fig. 6. Constraint evolution along the solution obtained by (16) using inconsistent initialization. Due to the large integration step,  $10^{-2}$ , some constraints are not satisfied on the order of the machine precision. Nevertheless, the constraint drift is not growing through the integration.

instantaneously from one to three. The intention here is to test (16) under this constraint singularity. The equation of motion is derived with six natural coordinates [53], which define the position of the moving joints,  $\mathbf{q} = [x_1, y_1, x_2, y_2, x_3, y_3]^T$ , and five constraints  $\Phi = [x_1^2 + y_1^2 - l^2, (x_2 - l)^2 + y_2^2 - l^2, (x_3 - 2l)^2 + y_3^2 - l^2, (x_2 - x_1)^2 + (y_2 - y_1)^2 - l^2, (x_3 - x_2)^2 + (y_3 - y_2)^2 - l^2]^T$ . The motion, started from a vertical rest position of the supporting links, is simulated over  $t \in [0, 100]$  s with  $10^{-2}$  time step. Due to the relatively large time step and long simulation time we do not expect a precise solution. Accordingly, compared to the exact numerical solution, the result reflected to  $x_3$  shows  $10^{-2}$  maximal error in the proposed solution and  $2 \times 10^{-4}$  error in the trusted solution. Note that, in numerical simulations with a time step,  $10^{-3}$ , the proposed and the trusted solution possessed the same accuracy level,  $\approx 10^{-7}$ , as that obtained in the previous examples. With the  $10^{-2}$  time step, integration of (17) with no constraint correction was unstable. The constrained equation (16), was also tested under inconsistent initialization. The results are depicted in Figs. 5 and 6.

5.4. Trajectory tracking control

Motivated by a recent development in trajectory tracking control [54], our intention is to show how (16) can be used to simulate dynamical systems which perfectly (rather than approximately) track a predefined reference trajectory. According to the classical philosophy of tracking control, we assume that the motion of the considered dynamical system is guided with kinematic constraints interpreted as control objectives. This view will allow us to embed all predefined reference trajectories in the constraint set (6). In the present context, this approach will allow perfect satisfaction of the control objectives without application of the Baumgartne’s constraint stabilization method as was used by Udwadia [54]. The idea is presented by simulating three “exercise” motions of a planar biped robot.

Consider a 7-link planar biped robot depicted in Fig. 7, with height  $L = 1.8$  m, mass  $M = 75$  kg and anthropometric geometric properties and mass distribution according to Table 1 [55]. The configuration of the biped is defined with nine absolute coordinates  $\mathbf{q} = [x, y, \theta, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6]^T$ . This coordinate set is independent for the “flying biped” while it becomes dependent if constraints are applied on the robot.

5.4.1. First exercise

Let us consider the biped restricted with nine kinematic constraints, specifically, six physical restrictions due to the ground

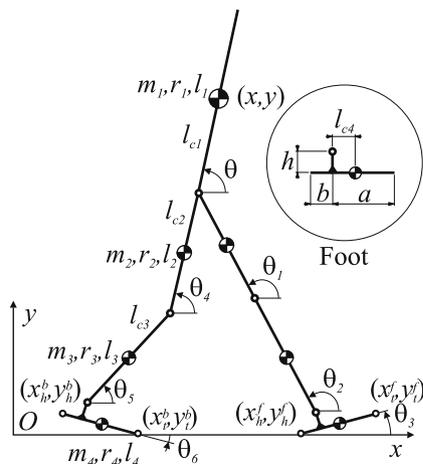


Fig. 7. Biped model with generalized coordinates and associated geometric and inertia properties. For each segment, the moment of inertial with respect to the center of mass is calculated as  $I_i = m_i r_i^2$ .

Table 1 Geometric and inertial parameters of the biped.

Description	No. (*)	$l_i/L$	$l_{c_i}/l_i$	$m_i/M$	$r_i/l_i$
Upper body	1	0.288	0.626	0.6780	0.496
Thigh	2	0.245	0.433	0.1000	0.323
Shank	3	0.246	0.433	0.0465	0.302
Foot	4	0.152	0.250	0.0145	0.475
Foot geometry		$a/l_4$	$b/l_4$	$h/L$	
		0.75	0.25	0.039	

contact of both feet, and three additional control objectives which specify the upper body angle and also define periodic flexion of both knees. The constraint set is given by:  $\Phi = [x_t^f - x_{td}^f, y_t^f, y_t^b, x_t^b - x_{td}^b, y_t^b, \theta - \theta_d, \varphi_k^f - \varphi_{kd}^f, \varphi_k^b - \varphi_{kd}^b]^T$ , where  $x_{td}^f = 0.507$ ,  $x_{td}^b = -0.097$  define the desired horizontal position for the toes on the forward and backward foot,  $\theta_d = 4\pi/9$  is the desired upper body angle,  $\varphi_k^f = \theta_1 - \theta_2$  is the relative angle at the forward knee while  $\varphi_{kd}^f = (\pi/6)(1 - \cos(\pi t))$  defines its desired motion,  $\varphi_k^b = \theta_4 - \theta_5$  is the relative angle at the backward knee with its desired motion defined by  $\varphi_{kd}^b = (\pi/10)(1 - \cos(\pi t))$ . The simulation result is depicted in Fig. 8.

5.4.2. Second exercise

Using the same biped model one can apply different constraints to generate a required motion. In this example, we apply four physical constraints which will hold the forward heel and the backward toe to remain on ground while the motion of the robot is dictated with five control constraints. Practically, we define the upper body angle  $\theta_d = 4\pi/9$ , the angular motion of the feet  $\theta_{3d} = (\pi/18)(\cos(\pi t) + 1)$ ,  $\theta_{6d} = -(\pi/18)(1 - \cos(\pi t))$  supported with periodic flexion of the backward knee  $\varphi_{kd}^b = (\pi/10)(1 - \cos(\pi t))$  and full extension of the forward leg. Accordingly, the constraint set is given by:  $\Phi = [x_h^f - x_{hd}^f, y_h^f, x_t^b - x_{td}^b, y_t^b, \theta - \theta_d, \theta_3 - \theta_{3d}, \theta_6 - \theta_{6d}, \varphi_k^b - \varphi_{kd}^b]^T$ , where  $x_{hd}^f = 0.262$ ,  $x_{td}^b = -0.038$ . A stroboscopic view of the simulated motion is depicted in Fig. 9.

5.4.3. Third exercise

The simulated motion here represents a balancing exercise with parallel legs while only the toes are on the ground. The constraint set is given by:  $\Phi = [x_t^f, y_t^f, x_t^b, y_t^b, \theta - \theta_d, \varphi_k^f - \varphi_{kd}^f, \varphi_k^b - \varphi_{kd}^b, \theta_3 - \theta_{3d}, \theta_6 - \theta_{6d}, x_{CoM}]^T$ , where  $\theta_d = 4\pi/9$ ,  $\varphi_{kd}^f = \varphi_{kd}^b = (\pi/3)(1 - \cos(2\pi t))$ .

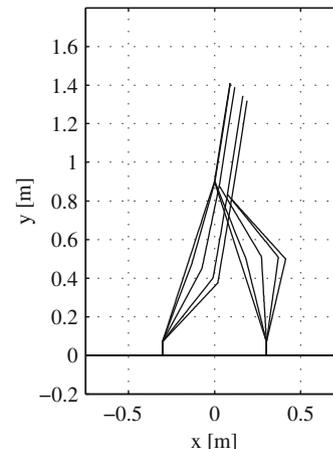
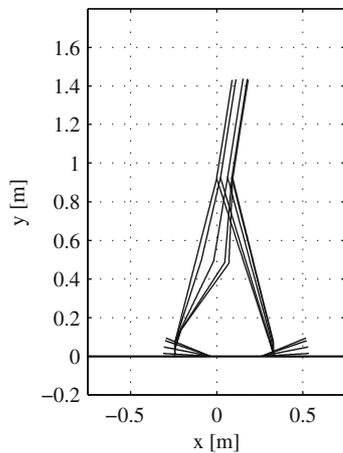
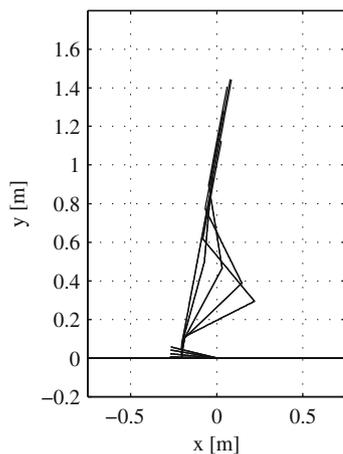


Fig. 8. The simulation is performed with  $10^{-2}$  time step over  $t \in [0, 100]$  s with  $\mathbf{q}(0) = [0.056, 1.220, 1.396, 1.920, 1.920, 0, 1.222, 1.222, 0]^T$  and  $\mathbf{v}(0) = \mathbf{0}$ . Along the motion, all physical and control constraints are satisfied up to  $5.2 \times 10^{-11}$ . The corresponding configurations in 50 successive depicted periods are overlapped.



**Fig. 9.** The simulation is performed with  $10^{-2}$  time step over  $t \in [0, 100]$  s with  $\mathbf{q}(0) = [0.056, 1.239, 1.396, 1.920, 1.920, 0.349, 1.292, 1.292, 0]^T$  and  $\mathbf{v}(0) = \mathbf{0}$ . The depicted 50 motion cycles show that the corresponding configurations in successive periods are overlapped. All constraints are satisfied up to  $2.7 \times 10^{-11}$ .



**Fig. 10.** The simulation is performed with  $10^{-2}$  time step over  $t \in [0, 100]$  s with  $\mathbf{q}(0) = [0.046, 1.252, 1.363, 1.363, 1.363, 0, 1.363, 1.363, 0]^T$  and  $\mathbf{v}(0) = \mathbf{0}$ . The stroboscopic view of the whole simulation, 20 cycles, shows that the configuration of the robot in successive periods are overlapped. All constraints are satisfied up to  $1.6 \times 10^{-12}$ .

$t/5$ ),  $\theta_{3d} = \theta_{6d} = -(\pi/27)(1 - \cos(2\pi t/5))$  while the center of mass of the biped is kept above the toes,  $x_{CoM} = 0$ . Note that the constraint set contains ten relations while the system is described with nine coordinates. In this light, regardless of the configuration of the biped, the constraint set is redundant at each time instant. Stroboscopic view of the balancing exercise is plotted in Fig. 10.

The reference trajectories in the above three simulations are selected such that the system cannot experience impacts (i.e., the necessary smoothness assumption required on  $\Phi$  is not violated). Using an appropriate impact resolution algorithm, the presented method could be applied to simulate nonsmooth motion such as bipedal walking. However, resolving impacts for multiple constraint dynamical systems is outside of the scope of this paper [27].

Finally, we would like to point out that by embedding a reference trajectory into the constraint set, we assumed that the reference motion could be enforced with an “ideal” control force which satisfies D’Alembert’s principle. This assumption, however, may generate a controllability issue on the trajectory tracking problem, as was pointed out by Blajer and Kolodziejczyk [56].

## 6. Conclusion

In order to simulate a constrained dynamical system, a governing equation (16) with embedded constraint correction terms is derived. This equation has a numerically stable implementation and allows the analyst to obtain a simulated solution over long time periods of constrained dynamical systems using simple generalized coordinates and standard ODE solvers. The presented formulation exploits the pseudoinverse of the constraint matrix, and as such, can also be used under dependent constraints and kinematic singularities. Although the idea is presented from the standpoint of classical mechanics, one can use it to simulate various physical systems modeled with differential-algebraic equations.

## References

- [1] J.L. Lagrange, *Mecanique Analytique*, Mme Ve Courcier, Paris, 1787.
- [2] C.F. Gauss, Über ein neues allgemeines grundgesetz der mechanik, *Zeitschrift für die Reine und Angewandte Mathematik* 4 (1829) 232–235.
- [3] G.A. Maggi, *Principii della Teoria Matematica del Movimento dei Corpi: Corso di Meccanica Razionale*, Ulrico Hoepli, Milano, 1896.
- [4] J.W. Gibbs, On the fundamental formulae of dynamics, *Am. J. Math.* 2 (1) (1879) 49–64.
- [5] P. Appell, Sur une forme generale des equations de la dynamique, *C.R. Acad. Sci. Paris* 129 (1899) 459–460.
- [6] T.R. Kane, D.A. Levinson, *Dynamics: Theory and Applications*, McGraw-Hill, New York, 1985.
- [7] F.E. Udwardia, R.E. Kalaba, A new perspective on constrained motion, *Proc. R. Soc. Lond. A* 439 (1992) 407–410.
- [8] F.E. Udwardia, R.E. Kalaba, *Analytical Dynamics: A New Approach*, Cambridge University Press, Cambridge, England, 1996.
- [9] F.E. Udwardia, R.E. Kalaba, On the foundations of analytical dynamics, *Int. J. Nonlinear Mech.* 37 (2002) 1079–1090.
- [10] L.A. Pars, *A Treatise on Analytical Dynamics*, John Wiley and Sons, New York, 1965.
- [11] Ju.I. Neimark, N.A. Fufaev, *Dynamics of nonholonomic systems*, AMS Providence, 1972.
- [12] F. Gantmacher, *Lectures in Analytical Mechanics*, Mir Publisher, Moscow, 1975 (English translation).
- [13] H. Goldstein, *Classical Mechanics*, Addison-Wesley, Reading, MA, 1980.
- [14] N.G. Chetaev, *Theoretical Mechanics*, Mir Publisher, Moscow, 1989 (English translation).
- [15] A.I. Lurie, *Analytical Mechanics*, Springer-Verlag, New York, 2002.
- [16] J. Baumgarte, Stabilization of constraints and integrals of motion in dynamical systems, *Comput. Method Appl. Mech. Engrg.* 1 (1972) 1–16.
- [17] C.W. Gear, B. Leimkuhler, G.K. Gupta, Automatic integration of Euler–Lagrange equations with constraints, *J. Comput. Appl. Math.* 12–13 (1985) 77–90.
- [18] P. Lötstedt, L. Petzold, Numerical solution of nonlinear differential equations with algebraic constraints. I: Convergence results for backward differentiation formulas, *Math. Comput.* 46 (174) (1986) 491–516.
- [19] C. Führer, B. Leimkuhler, Numerical solution of differential-algebraic equations for constrained mechanical motion, *Numer. Math.* 59 (1991) 55–69.
- [20] L.R. Petzold, Numerical solution of differential-algebraic equations in mechanical systems simulation, *Physica D* 60 (1–4) (1992) 269–279.
- [21] A.A. ten Dam, Stable numerical integration of dynamical systems subject to equality state-space constraints, *J. Engrg. Math.* 26 (1992) 315–337.
- [22] E. Eich, Convergence results for a coordinate projection method applied to mechanical systems with algebraic constraints, *SIAM J. Numer. Anal.* 30 (5) (1993) 1467–1482.
- [23] E. Bayo, R. Ledesma, Augmented lagrangian and mass-orthogonal projection methods for constrained multibody dynamics, *Nonlinear Dynam.* 9 (1–2) (1996) 113–130.
- [24] W. Blajer, Elimination of constraint violation and accuracy aspects in numerical simulation of multibody systems, *Multibody Syst. Dynam.* 7 (2002) 265–284.
- [25] 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 Trans. Robot.* 21 (5) (2005) 834–849.
- [26] W. Schiehlen, Multibody system dynamics: roots and perspectives, *Multibody Syst. Dynam.* 1 (1997) 149–188.
- [27] B. Brogliato, A.A. ten Dam, L. Paoli, F. Génot, M. Abadie, Numerical simulation of finite dimensional multibody nonsmooth mechanical systems, *ASME Appl. Mech. Rev.* 55 (2) (2002) 107–150.
- [28] J. D’Alembert, *Traite de Dynamique*, Paris, 1743.
- [29] J.J. Moreau, Quadratic programming in mechanics: dynamics of one-sided constraints, *J. SIAM Control* 4 (1) (1966) 153–158.
- [30] P. Lötstedt, Mechanical systems of rigid bodies subject to unilateral constraints, *SIAM J. Appl. Math.* 42 (2) (1982) 281–296.
- [31] A. Ben-Israel, T.N.E. Greville, *Generalized Inverse: Theory and Applications*, Springer, 2003.

- [32] C.W. Gear, Differential-algebraic equation index transformations, *SIAM J. Sci. Stat. Comput.* 9 (1988) 39–47.
- [33] L. Petzold, DASSL: A Differential/Algebraic System Solver. <<http://www.netlib.org/ode/ddassl.f>>.
- [34] K.E. Brenan, S.L. Campbell, L.R. Petzold, Numerical Solutions of Initial-Value Problems in Differential-Algebraic Equations, Elsevier Science, NY, 1989.
- [35] J.G. de Jalón, E. Bayo, Kinematic and Dynamic Simulation of Multibody Systems The Real-Time Challenge, Springer-Verlag, New York, 1994.
- [36] E. Hairer, G. Wanner, Solving Ordinary Differential Equations. II: Stiff and Differential-Algebraic Problems, second ed., Series in Computational Mathematics, Springer-Verlag, 1996.
- [37] G. Golub, C.V. Loan, Matrix Computations, third ed., The John Hopkins University Press, 1996.
- [38] B. Leimkuhler, L.R. Petzold, C.W. Gear, Approximation methods for the consistent initialization of differential-algebraic equations, *SIAM J. Numer. Anal.* 28 (1) (1991) 205–226.
- [39] P.E. Nikravesh, Initial condition correction in multibody dynamics, *Multibody Syst. Dynam.* 18 (2008) 107–115.
- [40] J.G. de Jalón, J. Unzueta, A. Avello, Natural coordinates for the computer analysis of multibody systems, *Comput. Method Appl. Mech. Engrg.* 59 (1986) 309–327.
- [41] C. Kraus, M. Winckler, H.G. Bock, Modeling mechanical DAE using natural coordinates, *Math. Comput. Model. Dynam. Syst.* 7 (2) (2001) 145–158.
- [42] E.J. Haug, Computer aided kinematics and dynamics of mechanical systems, Basic Methods, vol. I, Allyn and Bacon, Boston, 1989.
- [43] R. Serban, D. Negrut, E.J. Haug, F.A. Potra, A topology based approach for exploiting sparsity in multibody dynamics in cartesian formulation, *Mech. Struct. Mach.* 25 (3) (1997) 379–396.
- [44] M. Arnold, A. Fuchs, C. Führer, Efficient corrector iteration for DAE time integration in multibody dynamics, *Comput. Method Appl. Mech. Engrg.* 195 (50–51) (2006) 6958–6973.
- [45] A. Laulusa, O.A. Bauchau, Review of classical approaches for constraint enforcement in multibody systems, *J. Comput. Nonlinear Dynam.* 3 (2008) 011004.
- [46] O.A. Bauchau, A. Laulusa, Review of contemporary approaches for constraint enforcement in multibody systems, *J. Comput. Nonlinear Dynam.* 3 (2008) 011005.
- [47] J. Baumgarte, A new method of stabilization for holonomic constraints, *J. Appl. Mech.* 50 (1983) 869–870.
- [48] C.O. Chang, P.E. Nikravesh, An adaptive constraint violation stabilization method for dynamic analysis of mechanical systems, *J. Mech. Trans. Automat. Des.* 107 (1985) 488–492.
- [49] U.M. Ascher, H. Chin, S. Reich, Stabilization of DAEs and invariant manifolds, *Numer. Math.* 67 (1994) 131–149.
- [50] S.T. Lin, M.H. Hong, Stabilization method for numerical integration of multibody mechanical systems, *J. Mech. Des.* 120 (1998) 565–572.
- [51] W. Blajer, A geometrical interpretation and uniform matrix formulation of multibody system dynamics, *Z. Angew. Math. Mech.* 81 (4) (2001) 247–259.
- [52] R.A. Wehage, E.J. Haug, Generalized coordinate partitioning for dimension reduction in analysis of constrained dynamic systems, *J. Mech. Des.* 104 (1982) 247–255.
- [53] E. Bayo, A. Avello, Singularity free augmented lagrangian algorithms for constraint multibody dynamics, *Nonlinear Dynam.* 5 (1994) 209–231.
- [54] F.E. Udwarda, A new perspective on the tracking control of nonlinear structural and mechanical systems, *Proc. R. Soc. Lond. A* 459 (2003) 1783–1800.
- [55] D.A. Winter, Biomechanics and Motor Control of Human Movement, second ed., Wiley-Interscience, New York, 1990.
- [56] W. Blajer, K. Kołodziejczyk, A geometric approach to solving problems of control constraints: theory and a DAE framework, *Multibody Syst. Dynam.* 11 (2004) 343–364.