RESEARCH

Development of a constraint stabilization method of multibody systems based on fuzzy logic control

Ayman Nada[1] · Mona Bayoumi[2]

Abstract

The numerical solution of multibody systems is not a straightforward problem. The formulation of the equations of motion is augmented with the constraint equations that lead to a set of differential algebraic equations (DAEs). These constraints govern the relative motion between the system's components at the position level (geometric constraints) and may restrict the velocity of particular components (rolling constraints). There are several factors that determine the effectiveness of numerical integration methods and the extent of their applicability owing to the various motion circumstances. These factors include numerical stability throughout the integration and computation time, as well as allowable error percentage and the length of simulation time. In this regard, this research examines existing approaches for constraint stabilization during numerical integration and introduces a new methodology based on fuzzy control algorithm, whose coefficients are independent of the dynamic characteristics of different systems. Schematics of the new methodology are presented; two examples of spatial multibody systems with holonomic and nonholonomic constraints are solved to evaluate the effectiveness of the proposed method. It can be concluded that fuzzy control contributes an excellent solution for generic system configuration and is suitable for lengthy simulations with minimal computation time.

Keywords Multibody system dynamics · Baumgarte stabilization · Fuzzy logic control · Nonholonomic constraints

1 Introduction

1.1 Background

Dynamic modeling of multibody systems is a key aid in the analysis, design, optimization, and control of mechanical, aerospace, and mechatronic systems [2, 24, 41, 50]. During the

✉ A. Nada
  ayman.nada@ejust.edu.eg

  M. Bayoumi
  mona.elawa@bhit.bu.edu.eg

[1] School of Innovative Design Engineering, Faculty of Engineering, Egypt-Japan University of Science and Technology E-JUST, Alexandria 21934, Egypt

[2] Electrical Engineering Dept., Benha Faculty of Engineering, Benha University, Benha, Z.C. 13512, Benha, Qalubia, Egypt

past fifty years, various software tools, both commercial and used in scientific research, that enable engineers to construct realistic models of multibody systems have been developed [40, 46, 56]. Year after year, these tools grew more advanced, including methods for numerical integration, simulation, and mathematical transformation that aided the modelling of complicated systems. However, the significant amount of calculation time required for numerical calculations remains to be the first concern of the designer, hence the necessity of relying on computers with multiple cores and activating parallel computations to reduce this time. Reducing computation time is extremely important in the areas of design, parameter estimation, and model-based control methods.

In multibody system dynamics, the formulation of the equations of motion uses Newton–Euler approach augmented with the constraint equations that lead to a set of differential algebraic equations (DAEs) [8, 44, 45]. The numerical solution of the set of DAE is not a straightforward problem. There are several methods for solving the DAE. Among these methods, a stiff Gear algorithm was used, in which the equations of motion are substituted into the backward difference formula and solved simultaneously with the constraint equations that represent the kinematic joints of a system. This solution approach may lead to a worsening violation of the velocity level over time because it does not account for the error control on the velocity constraints [3, 16].

Real-time simulations are possible when computations are not excessively time-consuming and the iterations of the numerical solvers have to be sidestepped. Consequently, explicit numerical integration schemes with a deterministic timing behavior are crucial. One of the most popular and used methods to solve this problem consists of converting the DAE system into a set of ordinary differential equations (ODE) [47] by appending the second derivative with respect to time of the constraint equations [31, 36, 51, 54]. This way, the constraints are translated into the acceleration level; therefore, the equations of motion are then transformed from constrained differential equations to unconstrained differential equations at the position and velocity levels. In this method, the acceleration constraints are taken into consideration during the numerical integration of the equations of motion and, therefore, there is no violation at the acceleration level. On the other hand, the position and velocity constraint equations are ignored, leading to violation of the constraints at the position and velocity levels and the resulting "drift effect" in the given constraints [26].

To avoid the "drift effect," it is necessary to eliminate or at least minimize the violation of the constraints. There are three main ways to handle this problem: methods based on coordinate partitioning [31, 43, 52, 53], direct correction formulations [51, 54, 55], and approaches based on constraint stabilization [13–15, 34, 50].

1.2 Problem statement

The research work in this paper is an extension of the application of control theory to stabilizing the kinematic constraints of multibody systems by employing fuzzy control techniques. Fuzzy logic control (FLC) may not need to study the dynamic characteristics of the system in advance to figure out the control parameters. Instead, the control parameters are determined parametrically based upon the values of the variables (constraint violations) and the speed of their change. The challenge of introducing the FLC stabilization is to achieve a higher level of performance than other stabilization approaches by using an explicit integration method of any appropriate ODE solver with the largest possible step time of integration and the shortest possible calculation time.

1.3 Literature survey

The equations of motion of multibody systems are strongly nonlinear second-order ordinary differential equations in both the generalized coordinates and the Lagrange multipliers associated with the motion constraints. The numerical integration of these equations depends primarily on two factors. The first aspect pertains to the structure of equations, while the second aspect concerns the type of associated constraints [37].

The classical form of differential-algebraic equations of index-1, in which the generalized coordinates are represented by Cartesian coordinates that include both independent and dependent degrees of freedom, is the most prevalent [8]. Dealing with DAE index-1 permits easier calculation of forces associated with the constraints, which depend on the Lagrange multipliers [43]. On the other hand, although the Lagrange multipliers are of no interest, the formulation of equations of motion in terms of a minimal set of coordinates, typically in joint space, remains the method of choice for real-time and control applications. Nevertheless, the derivation of the equations is arduous and computationally complex due to the necessity of inverting the generalized mass matrix [17, 35].

Second, the motion constraints restrict the relative motions of the system's interconnected bodies. It may be given by algebraic equations that impose a definite relationship between the coordinates at the position level, called holonomic constraints. Holonomic constraints include the geometric descriptions of spherical, revolute, cylindrical, and translational joints. In addition, the constraints can be described by differential equations that restrict certain velocity components not originating from position constraints, called nonholonomic constraints [27]. Explicitly, the velocity constraint condition cannot be integrated in time to form a position constraint [12, 36, 39]. For a nonholonomic constraints, it is hard to figure out a closed-form geometric relationship. This means that the history of a constrained coordinate is needed to determine its current value. In this context, the rolling contact without slipping represents the most usual nonholonomic constraint, being linearly dependent on the generalized velocities [21, 42]. Chaplygin sleigh, the snakeboard, the rolling motion of discs and spheres, the skateboard, and the rattleback are all instances of nonholonomic systems that may be found in the literature [1, 42]. The transmission of motion through gear trains, using simplistic assumptions, could be considered an example of these nonholonomic constraints-subjugated systems [9, 31]. Each type of constraints associated with dynamic systems possesses a particular manipulation to minimize violations during numerical integrations.

A generalized coordinate partitioning method was introduced by Wehage [52]. The generalized coordinates are partitioned into independent and dependent coordinates. The independent coordinates are integrated, and the constraint equations are solved to get the dependent coordinates. The selection of a suitable set of independent coordinates is a pivotal aspect of the coordinate partitioning technique. Although this approach is theoretically rigorous, it may exhibit low numerical efficiency when adjusting the independent coordinate set, which constitutes one significant disadvantage of this method [28, 53]. Nada [31] proposed a strategy of selecting the independent coordinates in the case of systems subjected to nonholonomic constraints.

In direct correction algorithms, the numerical solution is projected back onto the constraints' manifold after each integration step. The key advantage of this approach lies in its ability to address constraint violations at both the position and velocity levels while maintaining the underlying dynamic equations of motion, independent of the integration algorithm utilized. In the case of holonomic and nonautonomous systems, however, this method is only formulated at the position level. Moreover, it must be noted that the kinematic constraints at the position level are typically nonlinear; consequently, the algorithm underlying

this method is an iterative numerical method, such as the Newton–Raphson method. Based on the computational tests conducted, this particular process requires a maximum of three iterations to effectively mitigate the violation of constraints at the position to an acceptable level. Conversely, since constraints at the velocity level are linear, the constraint violation for velocities is eliminated with a single step [7]. This could be a source of errors, and the constraint equations start to be violated. Hence, the approach is inappropriate for moderate or long simulations period [7, 26].

Among the constraint stabilization approaches, the Baumgarte stabilization method is the most attractive technique to overcome the drawbacks of the drift effect. Baumgarte's method can be looked upon as a PD type control to damp out the acceleration constraint violations by feeding back the violations of the position and velocity constraints. Moreover, PID type constraint stabilization method [10, 25] showed a smaller steady state error of the constraint equation than that using Baumgarte type method. The constraint stabilization approaches are probably the most popular due to their simplicity and easiness for computational implementation. However, their major drawback is the ambiguity in selecting the stabilization parameters, which ultimately can lead to failed simulations, even for systems that have valid solutions [22, 23, 26].

1.4 Scope and contribution

The present study focuses on improving the constraint stabilization methods by adding the robustness of the solution method. The suggested methodology entails the utilization of consistent ODE solvers for index-1 DAEs, independent of the dynamic characteristics of the system, the step time of numerical integration, and the type of constraints. Consequently, our aim is to employ fuzzy control to stabilize the constraints in that sense, compare the simulation results with well-established methods in the field, and evaluate the effectiveness of the suggested method by solving various application examples.

It can be concluded that the proposed method has the potential to serve as an effective alternative and a feasible option to provide a general and robust technique of constraint stabilization of multibody systems.

1.5 Organization of the paper

Following the introduction, the paper is organized in five sections. Section 2 is devoted to the construction of equations of motion using the multibody dynamics approach, and it discusses the direct correction and Baumgarte stabilization methods. Section 3 presents the proposed method of constraint stabilization using fuzzy logic control, outlines the solution method, and examines the method upon simple planar pendulum system. Section 4 illustrates the solution of spatial multibody systems subjected to holonomic constraints and provides comparison with other methods. Section 5 draws the updated scheme for multibody systems subjected to nonholonomic constraints. Finally, the concluding remarks are presented in Sect. 6.

2 Dynamics of multibody system

The multibody system consists of a number of bodies linked together by various types of joints that limit the relative motion between them. Figure 1 depicts the general configuration of multibody systems that may contain rigid and/or flexible bodies with small and/or large

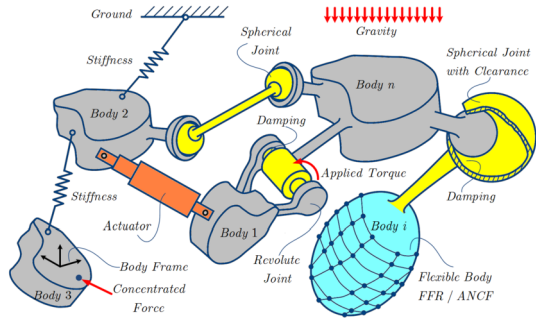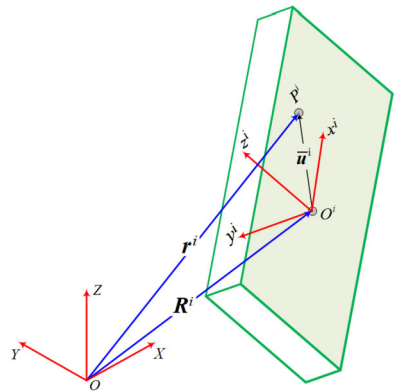**Fig. 1** General configuration of multibody systems

Fig. 2 Displacement field

deformations. These bodies are connected to each other using several types of joints and/or actuators and are subjected to several types of forces acting on them.

Defining a coordinate system (X, Y, Z) as the global frame that is fixed in time; the constraint equations of a holonomic system and its derivatives can be written in terms of the generalized coordinates and time as follows [44]:

$$\mathbf{C}^h(\mathbf{q}, t) = \mathbf{0}, \tag{1}$$

$$\dot{\mathbf{C}}^h(\mathbf{q}, \dot{\mathbf{q}}, t) = \mathbf{C}^h_{\mathbf{q}} \dot{\mathbf{q}} + \mathbf{C}^h_t = \mathbf{0}, \tag{2}$$

$$\ddot{\mathbf{C}}^h(\mathbf{q}, \dot{\mathbf{q}}, t) = \mathbf{C}^h_{\mathbf{q}} \ddot{\mathbf{q}} + \left(\mathbf{C}^h_{\mathbf{q}} \dot{\mathbf{q}}\right)_{\mathbf{q}} \dot{\mathbf{q}} + 2\mathbf{C}^h_{\mathbf{q}t} \dot{\mathbf{q}} + \mathbf{C}^h_{tt} = \mathbf{0}. \tag{3}$$

The vector $\mathbf{C}^h(\mathbf{q}, t)$ represents the kinematic constraints, $\mathbf{q}^i$ is the generalized coordinates vector, and t is the time. The matrix $\mathbf{C}^h_{\mathbf{q}} = \frac{\partial \mathbf{C}^h(\mathbf{q}, t)}{\partial \mathbf{q}}$ is the constraint Jacobian matrix. The vectors $\mathbf{C}^h_t$ and $\mathbf{C}^h_{tt}$ are the partial derivatives of the constraint function with respect to time.

The local coordinates of an arbitrary point located on body i with respect to body frame (x^i, y^i, z^i), see Fig. 2, can be described by the vector $\bar{\mathbf{u}}^i = \begin{bmatrix} \bar{u}^i_x & \bar{u}^i_y & \bar{u}^i_z \end{bmatrix}^T$, while the global position $\mathbf{r}$, i.e., with respect to the global frame, of that point can be expressed as

$$\mathbf{r}^i = \mathbf{R}^i + \mathbf{A}^i \, \bar{\mathbf{u}}^i, \tag{4}$$

where $\mathbf{r}^i = \begin{bmatrix} r^i_x & r^i_y & r^i_z \end{bmatrix}^T$ is the global position of an arbitrary point on body i, and the vector $\mathbf{R}^i = \begin{bmatrix} R^i_x & R^i_y & R^i_z \end{bmatrix}^T$ is the global position of the origin of the body coordinate

system. The matrix $\mathbf{A}^i = \mathbf{A}^i\left(\boldsymbol{\theta}^i\right)$ is the transformation matrix which depends on the orientational parameters $\boldsymbol{\theta}^i$. Therefore, Eq. (4) shows that the global position vector of an arbitrary point on a body can be written in terms of the generalized coordinates vector $\mathbf{q}^i$, where $\mathbf{q}^i = \begin{bmatrix} \mathbf{R}^{iT} & \boldsymbol{\theta}^{iT} \end{bmatrix}^T$. In the velocity analysis, it is assumed that the positions and orientations of the system are already known from the position analysis; differentiating Eq. (4) with respect to time, one obtains the global velocity vector $\dot{\mathbf{r}}$ as follows:

$$\dot{\mathbf{r}}^i = \begin{bmatrix} \mathbf{I} & -\mathbf{A}^i\,\widetilde{\mathbf{u}}^i\,\overline{\mathbf{G}}^i \end{bmatrix} \begin{bmatrix} \dot{\mathbf{R}}^i \\ \dot{\boldsymbol{\theta}}^i \end{bmatrix}, \tag{5}$$

where $\dot{\mathbf{R}}^i$ represents the global velocity of body frame, $\mathbf{I}$ is a 3×3 identity matrix, the generalized velocity vector $\dot{\mathbf{q}}^i = \begin{bmatrix} \dot{\mathbf{R}}^{iT} & \dot{\boldsymbol{\theta}}^{iT} \end{bmatrix}^T$. The matrix $\overline{\mathbf{G}}^i$ is defined such that $\overline{\boldsymbol{\omega}}^i = \overline{\mathbf{G}}^i\,\dot{\boldsymbol{\theta}}^i$, in which the velocity transformation matrix $\overline{\mathbf{G}}^i$ depends on the orientational parameters of the body, i.e., Euler angles or Euler parameters [4, 19, 48].

The equations of motion governing the dynamics of a multibody system can be obtained systematically by augmenting the corresponding matrices and vectors of all bodies, and can be expressed as follows [44]:

$$\begin{bmatrix} \mathbf{M} & \mathbf{C}_q^{hT} \\ \mathbf{C}_q^h & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}} \\ \boldsymbol{\lambda}^h \end{bmatrix} = \begin{bmatrix} \mathbf{Q} \\ \mathbf{Q}_d^h \end{bmatrix}, \tag{6}$$

where $\mathbf{M}$ is the assembled system mass matrix such that $\mathbf{M} = \mathrm{diag}\left(\mathbf{M}^i\right)$, the matrix $\mathbf{C}_\mathbf{q}^h$ is the Jacobian matrix, $\boldsymbol{\lambda}^h$ is the vector of Lagrange multipliers associated with the holonomic constraints. The vector $\mathbf{Q}_d^h$ is the vector that absorbs all quadratic terms of velocity. This vector can be obtained by solving Eq.(3) for $\mathbf{C}_\mathbf{q}^h\ddot{\mathbf{q}}$, one can conclude that

$$\mathbf{C}_\mathbf{q}^h\ddot{\mathbf{q}} = -\left(\mathbf{C}_\mathbf{q}^h\dot{\mathbf{q}}\right)_\mathbf{q}\dot{\mathbf{q}} - 2\mathbf{C}_{\mathbf{q}t}^h\dot{\mathbf{q}} - \mathbf{C}_{tt}^h = \mathbf{Q}_d^h. \tag{7}$$

The vector $\mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}, t)$ is the generalized force vector resulting from external forces $\mathbf{Q}_{\mathrm{ex}}$, as well as the centrifugal and Coriolis inertia forces $\mathbf{Q}_v$.

Equation (6) is linear in the accelerations and Lagrange multipliers. In most practical cases, the direct solution of Eq. (6) is preferable. Besides the conceptual simplicity of the method, it does not require the use of an iterative procedure and permits easier calculation of forces associated with the constraints, which depend on the Lagrange multipliers [43]. To get the acceleration vector and Lagrange multipliers, one has to get the inverse of the coefficient matrix, which can be obtained using lower–upper (LU) decomposition, singular value decomposition, and minimum norm least-squares solution of linear equation. Here is where the distinction between planar and spatial dynamic applications becomes more apparent due to the way of describing holonomic constraints and singularity concerns. In spatial dynamics, Euler parameters do not suffer from the kinematic singularity. However, the constraint forces associated with Euler parameter constraints are always zero and lead to a kinetic singularity, i.e., singular coefficient matrix. To overcome this difficulty, the mass matrix is extended as presented in [31]. On the other hand, Euler's angles lead to faster integration compared to the Euler parameters [4, 32] but suffer from kinematic singularities. Methods of avoiding the kinematic singularities associated with Euler angles utilization are described in Ref. [19].

It is noteworthy to mention that alternative techniques proposed in the literature do not necessitate constraint stabilization as they address the integration of the equations of motion

in the form of differential algebraic equations (DAE) [8]. Nevertheless, this topic is beyond the scope of this paper, thereby enabling the utilization of any appropriate ODE integration algorithm, obviating the necessity for a specialized DAE solver.

2.1 Direct correct formulation

In the numerical integration of the equations of motion, Eq. (6), only the differentiated form of the constraint function, Eq. (3) is satisfied and does not guarantee $\mathbf{C}^h(\mathbf{q}, t) = \mathbf{0}$ and $\dot{\mathbf{C}}^h(\mathbf{q}, \dot{\mathbf{q}}, t) = \mathbf{0}$ to be true as time integration goes on. Consequently, some constraints drifting at the position and velocity level may take place and encounter an integration errors. To counteract this drifting, it is necessary to correct the system states, after each integration step, by moving positions and velocities back to their manifolds $\mathbf{C}^h(\mathbf{q}, t)$, $\dot{\mathbf{C}}^h(\mathbf{q}, \dot{\mathbf{q}}, t)$ respectively, by adding small corrections. This correction process is called post stabilization procedure and can be implemented using the Newton–Raphson method [51].

2.1.1 Position stabilization

After each integration step, it is necessary to correct the systems state by moving positions back to their manifolds $\mathbf{C}^h(\mathbf{q}, t) = \mathbf{0}$ by means of small corrections. The stabilization step takes the result of the integration step as input and gives a correction so that the final result is closer to the constraint manifold. The position stabilization is then as follows (can be used using the Newton–Raphson method): Evaluate the Newton difference $\Delta\mathbf{q}_n$ and then update the generalized coordinate as follows:

$$\Delta\mathbf{q}_n = -\mathbf{C}_{\mathbf{q}}^{h\dagger}(\mathbf{q}_n)\,\mathbf{C}(\mathbf{q}_n) \tag{8}$$

$$\Updownarrow \tag{9}$$

$$\mathbf{q}_{n+1} = \mathbf{q}_n + \Delta\mathbf{q}_n. \tag{10}$$

Since the constraint Jacobian is a rectangular matrix, its inverse does not exist, and $\mathbf{C}_{\mathbf{q}}^{h\dagger}$ denotes the Moore–Penrose pseudoinverse of the matrix. The pseudoinverse computation is based on the singular value decomposition of the Jacobian matrix. Many algorithms have been developed in the literature to estimate the pseudoinverse of constraint Jacobian [54, 55]. When the error limit is obeyed, i.e., $\|\Delta\mathbf{q}_n\| \leq \epsilon$, then $\mathbf{q}(t) = \mathbf{q}_{n+1}$.

2.1.2 Velocity stabilization

The velocities can be moved back to their manifolds $\dot{\mathbf{C}}^h(\mathbf{q}, \dot{\mathbf{q}}, t)$ by means of one step correction, where no iterations are necessary [31], as follows:

$$\dot{\mathbf{q}}_{i+1} = \dot{\mathbf{q}}_i + \Delta\dot{\mathbf{q}}_i, \tag{11}$$

where the differences $\Delta\dot{\mathbf{q}}_i$ can be obtained by

$$\Delta\dot{\mathbf{q}}_i = -\mathbf{C}_q^{h\dagger}\dot{\mathbf{C}}(\mathbf{q}, t). \tag{12}$$

The vector $\dot{\mathbf{C}}^h(\mathbf{q}, \dot{\mathbf{q}}, t)$ can be obtained using Eq. (2).

The direct correct formulation described above is able to eliminate the violation of constraints at both the position and velocity levels without changing the equations of motion. In

some cases, the evaluation of the inverse matrix can exhibit numerical instabilities and can also be costly from the computational point of view. Moreover, neglecting the inertial properties of the systems can result in physical inconsistency, and different weighting factors should be utilized [6]. Some works have proposed including inertia of bodies as weighting factors, which allow for the adjustments to be made in an inverse manner to the system inertia [26].

2.2 Baumgarte stabilization

The second-order equations, Eq. (3), are unstable, and thus small perturbations due to numerical errors introduced by the integration process cannot be corrected naturally, and they only tend to be amplified. Baumgarte introduced feedback terms that penalize the system response if violations on the position or velocity constraint equations occur [5]. The Baumgarte stabilization method is to replace the differential equations Eq. (3) by

$$\ddot{\mathbf{C}}^h + 2\alpha\dot{\mathbf{C}}^h + \beta^2\mathbf{C}^h = \mathbf{0}, \tag{13}$$

where α and β are positive constants that weight the violations of the velocity and position constraint equations respectively and play the role of control terms. It should be noted that the method does not correct the constraint violations but simply keeps them under control. The use of the Baumgarte stabilization method is carried out by employing $\mathbf{C}_{\mathbf{q}}^h\ddot{\mathbf{q}} = \mathbf{Q}_d^h - 2\alpha\dot{\mathbf{C}}^h - \beta^2\mathbf{C}^h$ instead of Eq. (7) during the numerical integration process of the system equations of motion.

Baumgarte [5] highlighted that the suitable choice of the parameters α and β is performed by numerical experiments. In general, if α and β are chosen as positive constants, the stability of the general solution of Eq. (13) is guaranteed. When α is equal to β, critical damping is achieved, which usually stabilizes the system response more quickly. Neto et.al. [34] show that, for a multibody system made of rigid bodies, these constants have values in the range of $(1 \rightarrow 10)$. Several research articles have been published about how to set the gain and how that affects the system. Flores et.al. [15] pointed out that stabilizing the values of $\alpha = \beta = 5$ is a good choice for multibody systems consisting of rigid bodies that converge the constraints without oscillation. Nada [29] shows the values of α and β that give a damping factor of 0.707 to be $\alpha = 5$, $\beta = 12$.

The simple way for obtaining the Baumgarte parameters is to extend the constraint equation in Taylor's series while ignoring elements with orders greater than two [15]. In this case, Baumgarte parameters and time step may be stated mathematically as ($\alpha = 1/\delta t$, $\beta = \sqrt{2}/\delta t$).

Consequently, the Baumgarte method still has some ambiguity in determining optimal feedback gains. Indeed, it seems that the value of the parameters is purely empirical, and there is no reliable method for selecting the coefficients α and β. The improper choice of these coefficients can lead to unacceptable results in the dynamic simulation of multibody mechanical systems. Although Baumgarte's constraint stabilization gives good results in some applications, it does not help some particular configurations of the systems, such as in the neighborhood of kinematic singularities.

3 Constraint stabilization using fuzzy logic control

The fuzzy control methodology is known to many who are interested in the field of automatic control. In this paper, FLC will be applied to stabilize the constraints of multibody
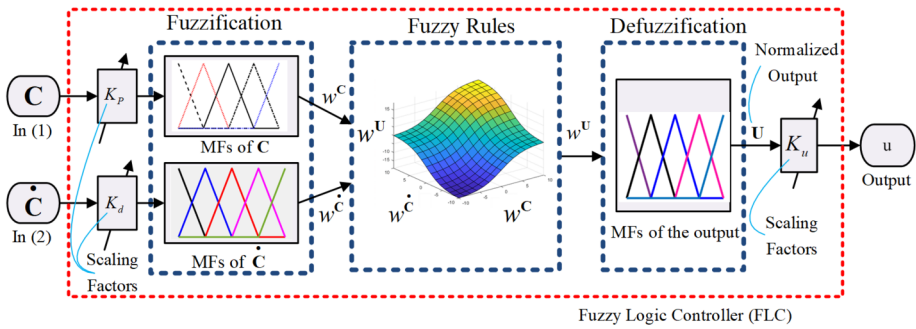
Fig. 3 Proposed FLC for constraint stabilization

systems. Therefore, it will be presented in a brief but simple manner to explain the utilization procedure, and the reader may refer to the fundamental references of FLC for further information [49]. In this section, the FLC is presented as a strategy oriented towards solving the shortcomings of other multibody system constraint stabilization methods.

3.1 Introduction to FLC

Fuzzy controllers take observable variables as inputs and, based on a set of rules, construct a compensation action as outputs. In the constraint stabilization of multibody systems, the inputs are the constraint function and its time derivative, i.e., $\mathbf{C}^h(\mathbf{q}, t)$ and $\dot{\mathbf{C}}^h(\mathbf{q}, t)$, the constraints at position and velocity levels. The entire FL controller is divided into mainly three different subsystems: fuzzification, fuzzy rule-based decision making, and defuzzification [38, 49]. In the fuzzification phase, the numerical input values are normalized, translated into linguistic variables, and then mapped into the degrees (probabilities) to which the inputs belong to respective fuzzy sets. These sets are described by certain membership functions (MFs), which may be defined using a variety of curves, although triangular or trapezoidal MFs are the most frequent. FLC's inputs $\mathbf{C}^h(\mathbf{q}, t)$ and $\dot{\mathbf{C}}^h(\mathbf{q}, \dot{\mathbf{q}}, t)$ for constraint stabilization problem are normalized, according to the permissible violations of constraints, using the scaling factors $\left(K_p \text{ and } K_d\right)$ such that they fall within a specific range, the most typical range is $[-1, 1]$. The output of the fuzzification phase is the probabilities of each MF for every input $A_{\text{input}}^{\text{MFs}}$.

The fuzzy rule table is a matrix of values that defines what the output control surface should look like. The number of inputs and amount of inputs along with the number of fuzzy membership functions will determine the rule table dimensions. The number of elements that are placed in the rule table indicates the number of membership functions used in the fuzzy system. The output of the fuzzy rules table is the strength of each rule based on the inputs' probabilities. The defuzzification phase, called inference, utilizes fuzzy logic (FL) operators to map the function between input and output. The two basic methods of fuzzy inference are Mamdani and Sugeno [49]. The fuzzy inferences used with the Mamdani form of fuzzy rules includes both Mamdani and Gödelian combinations. The first includes the minimum and product inferences, whereas the second includes Dienes-Rescher, Lukasiewicz, and Zadeh inferences [49]. On the other hand, Sugeno shows that both zeroth-order and higher-order forms can be implemented. Finally, the gain factor K_c for the output is determined in such a way that the output of the FLC can generate the required control action. The schematic of the proposed FL controller is shown in Fig. 3.
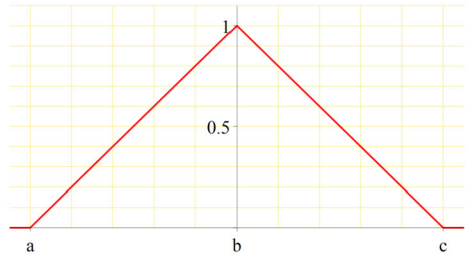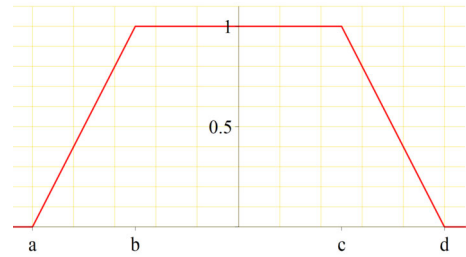
Fig. 4 Triangular MF

Fig. 5 Trapezoidal MF

3.1.1 Fuzzification

A membership function (MF) is a function that associates each point in X, where $x \in X$, with a real integer in the range $[0, 1]$. There are several MFs which can be used as fuzzy sets, for example, the mathematical representation for triangular MF, see Fig. 4, is defined by its parameters $\{a, b, c\}$ such that

$$A(x) = \begin{cases} 0 & x \leq a, \\ \frac{x-a}{b-a} & a \leq x \leq b, \\ \frac{c-x}{c-b} & b \leq x \leq c, \\ 0 & x \geq c. \end{cases} \tag{14}$$

$A(x)$ is the probability of the input x, and the parameters $\{a, b, c\}$ with $a < b < c$ determine the x-coordinates of the three corners of the underlying triangular MF. Depending on the relationships between a, b, and c, triangular MFs may be asymmetric. A trapezoidal MF specified by four parameters $\{a, b, c, d\}$, see Fig. 5, is defined as

$$A(x) = \begin{cases} \frac{x-a}{b-a} & a \leq x \leq b, \\ 1 & b \leq x \leq c, \\ \frac{d-x}{d-c} & c \leq x \leq d, \\ 0 & x \leq a \cup x \geq d. \end{cases} \tag{15}$$

The singleton MF is defined by its parameter c such as

$$A(x) = \begin{cases} 1 & x = c, \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

To reduce computation time, the MFs can be represented using logic functions based on logic operations, which can be built in combination with the fixing of chosen geometrical parameters that do not affect the subsequent control operations.

3.1.2 Inference engine and defuzzification

In the rule base unit, there are a set of rules that relate to input and output variables of the controller. These rules are simple if-then structures with a condition and appropriate value of each rule. For example, rule 1: if $\mathbf{C}^h$ is $(-\Downarrow)$ and $\dot{\mathbf{C}}^h$ is $(-\Downarrow)$, then the appropriate value from the rule table c^1 is -1, where $(-\Downarrow)$ means Negative Big. Table 1 shows the appropriate value of each rule, i.e., $c^i s$, where $i = 1 \longrightarrow N$ and N is the number of rules.

In the case of manipulating two inputs, one can calculate the measure of the influence of rule i on the output by estimating the rule-strength w^i according to the corresponding probabilities as follows:

$$w^i = A_1^j \cap A_2^k, \qquad (j, k = 1 \longrightarrow 5), \tag{17}$$

where A_1^j is the probability of jth MF for input 1, and A_2^k is the probability of MF k for input 2. The fuzzy OR/AND operator simply selects the maximum/minimum of the two probabilities.

The inference engine manages the input variables and the rule base, and it decides which appropriate values are used. There are two types of inference systems varying somewhat in the way outputs are determined: Mamdani-type and Sugeno-type [49]. Mamdani's fuzzy inference method expects the output's MFs to be fuzzy sets that needed defuzzification.

Instead of integrating across the two-dimensional function to find the centroid, Sugeno-type systems use a single spike as the output MF rather than a distributed fuzzy set. This type of output is known as a singleton output MF, Eq. (16). It enhances the efficiency of the defuzzification process because it greatly simplifies the computation required by the Mamdani method. In this case, the weighted average of a few data points is used to calculate the control output. A typical rule in a Sugeno fuzzy model for two-input single-output has the form: If input 1 is x and input 2 is y, then output is $z = ax + by + c$, and for zero-order Sugeno model, the output level z is a constant ($a = b = 0$), i.e., $z = c$. Thus, the final output of the system is the weighted average of all rule outputs, computed as

$$\mathbf{U} = \sum_{i=1}^{N} w^i z^i \bigg/ \sum_{i=1}^{N} w^i, \tag{18}$$

where w^i is computed by using Eq. (17), which is the least significant bits of the inputs, z^i is the rule-table value (zero-order Sugeno model), and N is the number of rules according to Table 1. The weighted average method is suitable for symmetric membership functions. The defuzzification stage estimates the final output according to Eq. (18). Note that the definition of the outputs is only a singleton MF; by doing this the computation of the centroid of mass or any other defuzzification method is avoided because there is only one value defined for the output.

3.2 FLC constraint stabilization

The proposed method, introduced in this paper, can be presented as shown in Fig. 6, which can be viewed as a regulator problem of a double integrator system. The "drift effect" due

Table 1 5×5 rule-base table. $(-\Downarrow)$ Negative Big, $(-\downarrow)$ Negative Small, $(\longleftrightarrow)$ Zero, $(+\uparrow)$ Positive Small, $(+\Uparrow)$ Positive Big

$C/\dot{C}$	$-\Downarrow$	$-\downarrow$	$\longleftrightarrow$	$+\uparrow$	$+\Uparrow$
$-\Downarrow$	-1	-1	-1	$-1/2$	0
$-\downarrow$	-1	-1	$-1/2$	0	$1/2$
$\longleftrightarrow$	-1	$-1/2$	0	$1/2$	1
$+\uparrow$	$-1/2$	0	$1/2$	1	1
$+\Uparrow$	0	$+1/2$	1	1	1

Fig. 6 Constraint stabilization using FL controller
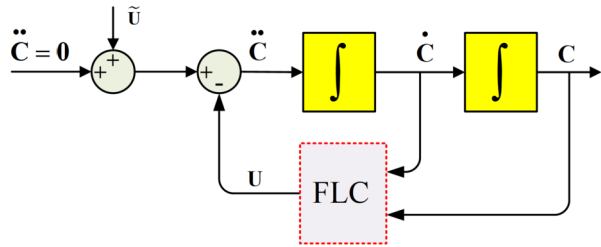
to violation of the constraints might be described as unknown disturbances to the system $\tilde{\mathbf{U}}$. The control action of the FLC needs to reduce the impact of the unknown disturbances as much as possible. These disturbances vary in accordance with the method of numerical integration that is being used (whether it be explicit or implicit), the time step and interval of integration, as well as the allowable absolute and relative error of integration. When developing generic subroutines that deal with multibody systems, it is impossible to predict the values the user would choose since there are so many possible combinations. As a result, the system is considered fuzzy, and perhaps the difference is that the rate of change of the error is not calculated analytically; rather, it may be computed numerically using Eq. (2), which helps to reduce calculation time.

The paper suggests implementing Sugeno-type systems to estimate the output of the FL controller. This is due to the fact that it improves the efficiency of the defuzzification process by considerably simplifying the calculation needed by the Mamdani approach. The controller input variables are the components of $\mathbf{C}^h(\mathbf{q}, t)$ and $\dot{\mathbf{C}}^h(\mathbf{q}, \dot{\mathbf{q}}, t)$, i.e., the constraint functions in the position and velocity levels.

Five MFs subjected to each input will be utilized in the fuzzification phase. Each membership MF_1 to MF_5 describes the probability of the input variable as $(-\Downarrow)$ Negative Big, $(-\downarrow)$ Negative Small, $(\longleftrightarrow)$ Zero, $(+\uparrow)$ Positive Small, and $(+\Uparrow)$ Positive Big. Three fuzzy sets of inputs, illustrated in Figs. 7, 8, 9, will be investigated to demonstrate the robustness of the numerical integration with each set with different user's preferences. Set *I* is comprised of triangular MFs, set *II* of trapezoidal MFs, and set *III* of Gaussian MFs.

Inference engine consists of 25 rules, practically, the 5×5 rule-base table, see Table 1 will be implemented. Zeroth-order Sugeno form of fuzzy rules is used, and the resulting inference output surface is represented Fig. 10 for set *I*. Singleton MFs that are assigned for output variable are shown in Fig. 11.

As shown in Fig. 6, the constraint violation is introduced to the system as unknown disturbances $\tilde{\mathbf{U}}$, and thus

$$\ddot{\mathbf{C}}^h(\mathbf{q}, \dot{\mathbf{q}}, t) + \tilde{\mathbf{U}}^h = \mathbf{0}. \tag{19}$$

Fig. 7 Set I: Triangle MFs for input variables
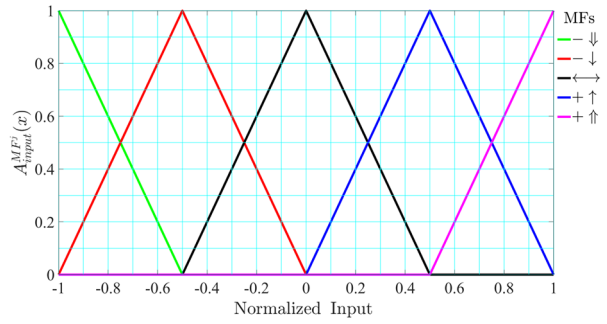

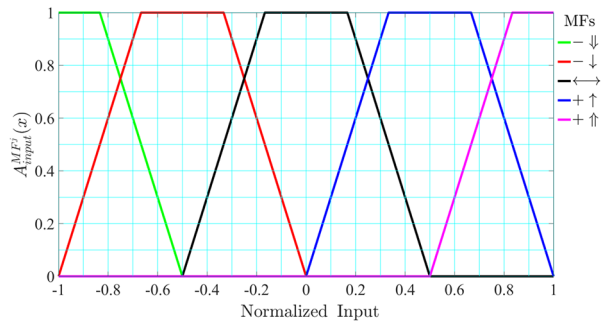**Fig. 8** Set II: Trapezoidal MFs for input variables
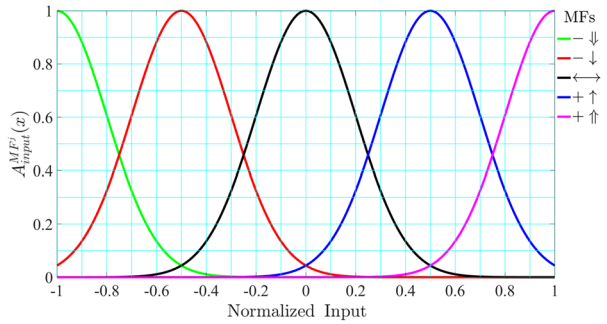

Fig. 9 Set III: Gauss MFs for input variables

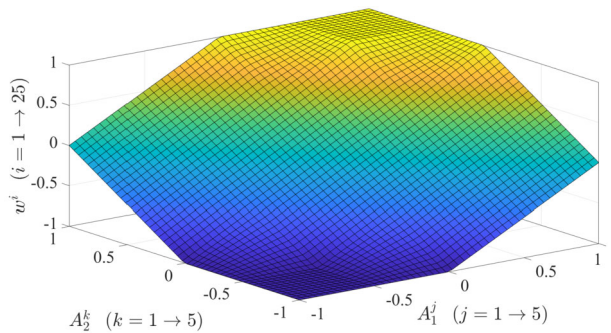**Fig. 10** Standard control surface constructed from set I

Fig. 11 Singleton MFs for output
variable
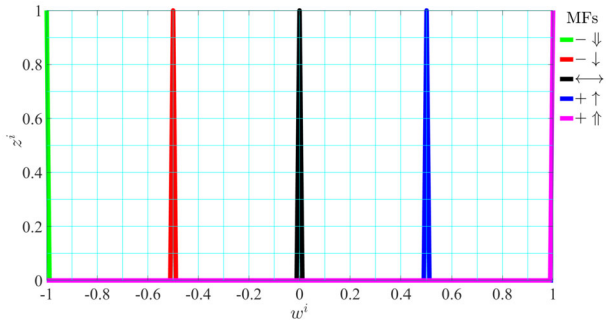
The FLC action $\mathbf{U}^h\left(\mathbf{C}^h, \dot{\mathbf{C}}^h, t\right)$ must compensate this drifting, i.e., $\mathbf{U}^h = \tilde{\mathbf{U}}^h$. Thus differential equations Eq. (3) can be represented as

$$\ddot{\mathbf{C}}^h(\mathbf{q}, \dot{\mathbf{q}}, t) + \mathbf{U}^h = \mathbf{C}^h_{\mathbf{q}}\ddot{\mathbf{q}} - \mathbf{Q}_d + \mathbf{U}^h = \mathbf{0}. \tag{20}$$

Therefore, the right-hand side of Eq. (7) is modified to

$$\mathbf{C}^h_{\mathbf{q}}\ddot{\mathbf{q}} = \mathbf{Q}^h_d - \mathbf{U}^h. \tag{21}$$

3.3 Outline of solution method

This section outlines the proposed method of constraint stabilization for multibody system through the utilization of the fuzzy logic control, as described in the last section. The outline can be summarized into the following steps:

1. Formulate the equations of motion of the system by defining the mass matrix, Coriolis inertia forces, and external forces.
2. Based on the initial configuration of the system, construct the constraint functions, its Jacobian, and the quadratic velocity vector.
3. Select the suitable membership functions for the constraint stabilization using FLC. Set *III* can be used as an initial selection for the control inputs, and a singleton membership function for the controller output.
4. Based on the permissible constraint violations, select the scaling factors K_p, K_d, and K_u. Initial selection can be carried out to normalize the norm-1 of the constraint functions and its time derivatives to desired values, such that $K_p = 1/\|\mathbf{C}\|$, $K_d = 1/\|\dot{\mathbf{C}}\|$, and $K_u = 100K_p$. In this case, it is possible to set upper boundaries on violations, thereby enabling their regulation. In general, the values of the scaling constants can be determined based on the violations subsequent to the first integration steps.
5. Calculate the control action and stabilize the constraints by using Eq. (21).
6. Solve Eq. (6) for the acceleration vector and Lagrange multipliers.
7. Integrate the generalized accelerations using any appropriate ODE solver, the initial selection of ODE45 can be utilized.

Consider a simple example of triple pendulum rotating in the xy plane, see Fig. 12. The system is composed of three rigid bodies and three revolute joints. The point O is the location of the revolute joint that imposes a connection between the ground and the first body. The points P^1, P^2 are the locations of revolute joints acting as connections between the three bodies. The pendulum possess identical geometric dimensions, having a

Fig. 12 Planar model of triple
pendulum
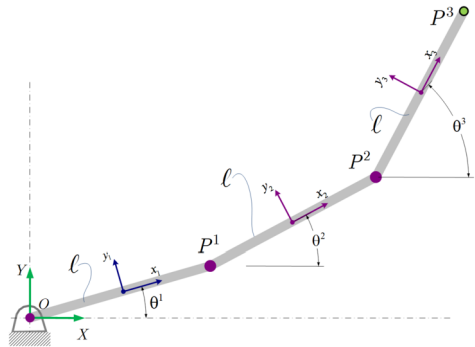
length of $l = 0.3$ [m], a mass of $m = 0.25$ [Kg], and a moment of inertia about the centroidal axis $J = \frac{1}{12} \, ml^2$ [Kg.m^2]. The body frames are attached to the centers of masses, i.e., $\mathbf{Q}_v = \mathbf{0}$. The pendulum is subjected to gravitational loading toward the y-axis. The mass matrix can be constructed as $\mathbf{M} = \text{diag}\left(\mathbf{M}^i\right)$, where $\mathbf{M}^i = \text{diag}\left(\begin{bmatrix} m^i & m^i & J^i \end{bmatrix}\right)$. The generalized force can be constructed as $\mathbf{Q}_{\text{ex}} = \begin{bmatrix} \mathbf{Q}_g^1 & \mathbf{Q}_g^2 & \mathbf{Q}_g^3 \end{bmatrix}^T$, where $\mathbf{Q}_g^i = \begin{bmatrix} 0 & m^i g & 0 \end{bmatrix}^T$, $g = -9.81$ [m/s^2] is the acceleration constant. The constraints can be written as follows:

$$\mathbf{C} = \begin{bmatrix} \mathbf{R}^1 + \mathbf{A}^1\bar{\mathbf{u}}^O \\ \mathbf{R}^1 + \mathbf{A}^1\bar{\mathbf{u}}^{1P1} - \mathbf{R}^2 + \mathbf{A}^2\bar{\mathbf{u}}^{2P1} \\ \mathbf{R}^2 + \mathbf{A}^2\bar{\mathbf{u}}^{2P2} - \mathbf{R}^3 + \mathbf{A}^3\bar{\mathbf{u}}^{3P3} \end{bmatrix}, \text{where} \mathbf{A}^i = \begin{bmatrix} \cos\theta^i & -\sin\theta^i \\ \sin\theta^i & \cos\theta^i \end{bmatrix},$$

where $\mathbf{R}^i = \begin{bmatrix} R_x & R_y \end{bmatrix}^{iT}$, $\bar{\mathbf{u}}^O = \begin{bmatrix} -l/2 & 0 \end{bmatrix}^T$, $\bar{\mathbf{u}}^{1P1} = \begin{bmatrix} l/2 & 0 \end{bmatrix}^T$, $\bar{\mathbf{u}}^{2P2} = \begin{bmatrix} -l/2 & 0 \end{bmatrix}^T$, $\bar{\mathbf{u}}^{3P2} = \begin{bmatrix} l/2 & 0 \end{bmatrix}^T$. The membership functions can be constructed in the preprocessing stage by using the MATLAB functions *addMF*, *addRule*, and *evalfis* as

```
% MF for input1
MbsdFis = addMF(MbsdFis,'C','gaussmf',[0.2 -1],'Name','BN');
.
.
MbsdFis = addMF(MbsdFis,'C','gaussmf',[0.2 1],'Name','BP');
% MF for input2
MbsdFis = addMF(MbsdFis,'Cd','gaussmf',[0.2 -1],'Name','BN');
.
.
MbsdFis = addMF(MbsdFis,'Cd','gaussmf',[0.2 1],'Name','BP');
% MF for output1
MbsdFis = addMF(MbsdFis,'output','constant',[-1],'Name','BN');
.
.
MbsdFis = addMF(MbsdFis,'output','constant',[1],'Name','BP');
%rules
rule1= "err==BN & derr==BN => output=BN" ;
.
.
rule25= "err==BP & derr==BP => output=BP" ;
MbsdFis=addRule(MbsdFis,rules);
```
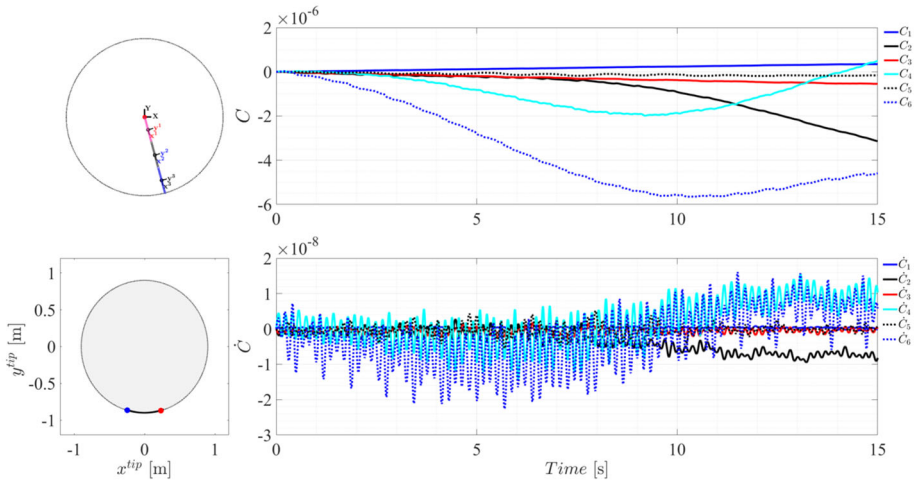
Fig. 13 Direct correct stabilization of triple pendulum in xy-plane - small displacement

During the integration process, the $\mathbf{Q}_d$ can be corrected, as Eq. (21), by using the following command:

```
Uh=Ko*evalfis(MbsdFis,[Kp*C', Kd*Cq*qd],MbsdFisOptions).
```

The system of triple pendulum exhibits chaotic behavior, indicating extreme sensitivity to the initial conditions. When the motion stays close to the equilibrium position, i.e., the initial displacement is small, it undergoes simple harmonic motion. Nevertheless, in cases where the initial displacement is large, repeating oscillations turn into chaotic motion. Solving the equations of motion demonstrates this behavior. It can be observed that the implementation of the direct correct stabilization method results in fast numerical integration with minimal violations, particularly in scenarios involving small displacement and short simulation time, as depicted in Fig. 13. The situation becomes significantly worse in the case of large displacement, resulting in a chaotic scenario, and lengthy simulation time. Figure 14(✓) shows wrong results as the tip point trajectory goes outside the reach of the pendulum. It is clear that violations of constraints increase over time as integration progresses. Figure 15 shows the effectiveness of the proposed FLC constraint stabilization method. The method allows the user to regulate the violations to upper boundaries and enables simulation for longer periods of time. In this example, the upper limits of the constraints were placed on the position level at 1e−6 and on the velocity level at 1e−4, respectively. In the rest of the paper, a comprehensive evaluation will be conducted to compare all the methodologies introduced upon their implementation on two distinct spatial systems.

4 Example I: multibody system with holonomic constraints

In this section, the rotary-pendulum shown in Fig. 16 is used to examine constraint stabilization methods. The prototype consists of an L-shape arm that is connected to a revolute shaft and rotates between ±180° degrees. At the end of the arm, a pendulum body is suspended on a horizontal axis at the end of the L-shape arm. The pendulum link and weight combined
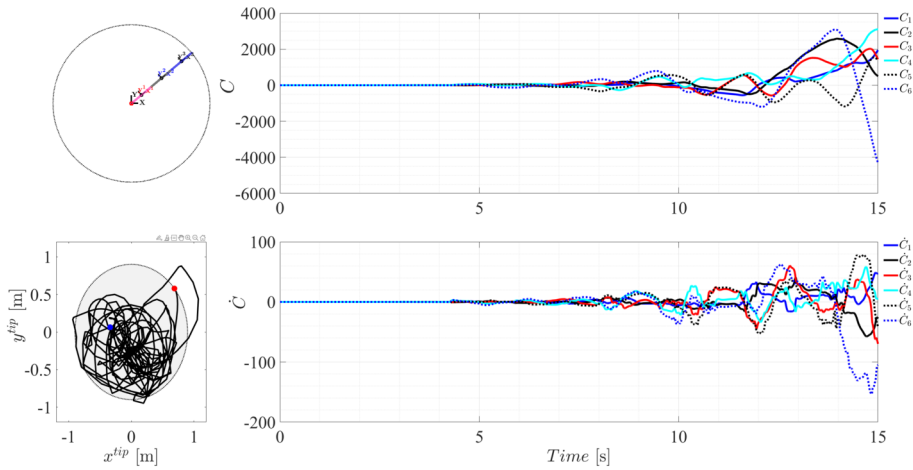
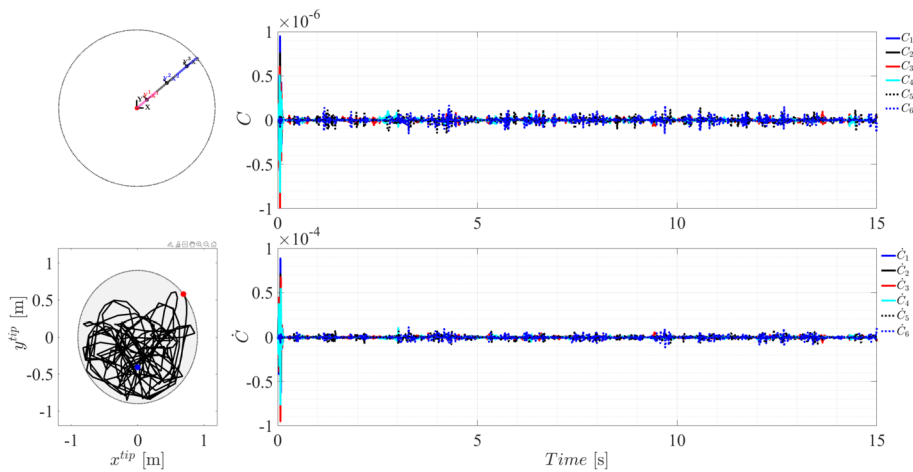**Fig. 14** Direct correct stabilization of triple pendulum in xy-plane - large displacement

Fig. 15 FLC stabilization of triple pendulum in xy-plane - large displacement. ($\searchow$) initial configuration, ($\nearrow$) **C**, ($\swarrow$) trajectory of the tip point, and ($\searrow$) **Ċ**

have the mass M_p and a total length of L_p and can rotate freely in the vertical plane. The length from arm pivot to pendulum pivot is r_a, and the length from pendulum center of mass to its pivot is l_p.

The pendulum and arm angle are measured by encoders attached. The measured variables are the angular displacements and velocities of the pendulum and of the arm. The system is interfaced by means of a data acquisition card, and the measurements are recorded. The multibody model of the system is constructed and validated in [30, 32]. This pre-evaluated example will be used here to examine the effectiveness of constraint stabilization methods.

As a multibody system, the model consists of two unsymmetrical bodies. The body's frames are fixed on the beginning of each body (not on the centers of masses), and they can rotate on two perpendicular axes through rotational joints. The generalized coordinates
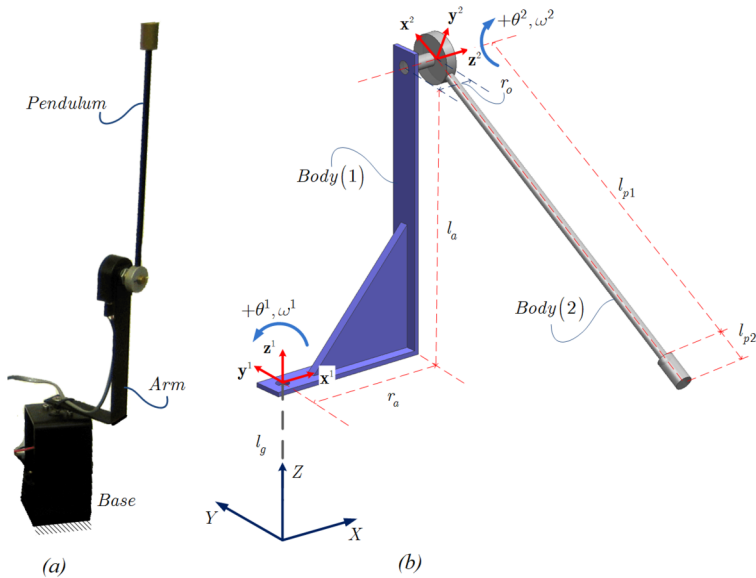
Fig. 16 Multibody model of spatial pendulum

of the system include the Cartezian positions of the frames and the Euler angles of each body, such that $\mathbf{q}^i = \begin{bmatrix} R_x^i & R_y^i & R_z^i & \phi^i & \theta^i & \psi^i \end{bmatrix}^T$, $i = 1, 2$. At the configuration shown in Fig. 16, the frame of body 1, the L-shape arm, $(\mathbf{x}^1, \mathbf{y}^1, \mathbf{z}^1)$ is parallel to the global frame $(\mathbf{x}^0, \mathbf{y}^0, \mathbf{z}^0)$, and the axis of rotation is the global z-axis that coincides with the axis $\mathbf{z}^1$. The constraint functions of the revolute joint between the ground, body 0, and body 1 can be expressed as

$$\mathbf{C}^{GR_1} = \begin{bmatrix} \mathbf{r}^{1P} - \mathbf{r}^{0P} \\ \mathbf{z}^{1T} \cdot \mathbf{x}^0 \\ \mathbf{z}^{1T} \cdot \mathbf{y}^0 \end{bmatrix} = \begin{bmatrix} R_1^1 \\ R_2^1 \\ R_3^1 - l_g \\ \sin\phi^1 \sin\theta^1 \\ -\cos\phi^1 \sin\theta^1 \end{bmatrix}_{5 \times 1} = \mathbf{0}. \tag{22}$$

Note that $\mathbf{r}^{0P} = \mathbf{R}^0 + \mathbf{A}^0 \bar{\mathbf{u}}^0$, $\mathbf{R}^0 = \mathbf{0}$, $\mathbf{A}^0 = \mathbf{I}$, $\bar{\mathbf{u}}^{0P} = \begin{bmatrix} 0 & 0 & l_g \end{bmatrix}^T \iff \mathbf{r}^{0P} = \bar{\mathbf{u}}^{0P}$, $\mathbf{x}^0 = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T$, $\mathbf{y}^0 = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$, and other terms regarding the L-shape body are $\mathbf{r}^{1P} = \mathbf{R}^1 + \mathbf{A}^1 \bar{\mathbf{u}}^{1P}$, since $\bar{\mathbf{u}}^{1P} = \mathbf{0} \iff \mathbf{r}^{1P} = \mathbf{R}^1$, where the rotation axis $\mathbf{z}^1$ can be found as the third column of the transformation matrix as follows:

$$\mathbf{z}^1 = \mathbf{A}_{(:,3)}^1 = \begin{bmatrix} \sin\phi^1 \sin\theta^1 \\ -\cos\phi^1 \sin\theta^1 \\ \cos\theta^1 \end{bmatrix} \tag{23}$$

such that the last two equations in Eq. (22) are the first and second elements of $\mathbf{z}^1$. Similarly, the constraint functions of the revolute joint between the L-shape body and the pendulum

can be expressed as

$$\mathbf{C}^{GR_2} = \begin{bmatrix} \mathbf{r}^{1K} - \mathbf{r}^{2K} \\ \mathbf{z}^{2T} \cdot \mathbf{x}^1 \\ \mathbf{z}^{2T} \cdot \mathbf{y}^1 \end{bmatrix} = \mathbf{0}, \tag{24}$$

where $\bar{\mathbf{u}}^{1K} = \begin{bmatrix} r_a & 0 & l_a \end{bmatrix}^T$, $\bar{\mathbf{u}}^{2K} = \begin{bmatrix} 0 & 0 & -r_o \end{bmatrix}^T$, $\mathbf{x}^1 = \mathbf{A}^1_{(:,1)}$, $\mathbf{y}^1 = \mathbf{A}^1_{(:,2)}$, and $\mathbf{z}^2 = \mathbf{A}^2_{(:,3)}$, Eq. (24) can be written as

$\mathbf{C}^{GR_2} =$

$$\begin{bmatrix} R_x^1 + \left(\cos\phi^1\cos\psi^1 - \sin\phi^1\cos\theta^1\sin\psi^1\right)r_a + l_a\sin\phi^1\sin\theta^1 - R_x^2 + r_o\sin\phi^1\sin\theta^2 \\ R_y^1 + \left(\sin\phi^1\cos\psi^1 + \cos\phi^1\cos\theta^1\sin\psi^1\right)r_a - l_a\cos\phi^1\sin\theta^1 - R_y^2 - r_o\cos\phi^1\sin\theta^2 \\ R_z^1 + \left(\sin\theta^1\sin\psi^1\right)r_a + \cos\theta^1 l_a - R_z^2 + r_o\cos\theta^2 \\ \sin\phi^1\sin\theta^1\sin\phi^2\sin\theta^2 + \cos\phi^1\sin\theta^1\cos\phi^2\sin\theta^2 + \cos\theta^1\cos\theta^2 \\ \left(-\cos\phi^1\sin\psi^1 - \sin\phi^1\cos\theta^1\cos\psi^1\right)\sin\phi^2\sin\theta^2 - \cdots \\ \cdots\left(-\sin\phi^1\sin\psi^1 + \cos\phi^1\cos\theta^1\cos\psi^1\right)\cos\phi^2\sin\theta^2 + \sin\theta^1\cos\psi^1\cos\theta^2 \end{bmatrix}_{5\times1}$$

$$= \mathbf{0}.$$

There are ten constraint functions that represent the holonomic nature of a fully nonlinear and nonautonomous system. The system constraint functions can be assembled as

$$\mathbf{C_q}\left(\mathbf{q}\right) = \begin{bmatrix} \mathbf{C}^{GR_1} \\ \mathbf{C}^{GR_2} \end{bmatrix}_{10\times1} = \mathbf{0}. \tag{25}$$

Accordingly, the model is validated in [30, 32] as the pendulum released for free oscillation from a vertical position at a starting angle of 170 degrees, and the angular velocities of the arm and pendulum, i.e., $\left(\bar{\omega}^1, \bar{\omega}^2\right)$ are recorded, where $\bar{\omega}^i$ is the local angular velocity of body i. This validation of the multibody model enables us to choose the method of integration and define its capabilities.

Four distinct integrators are utilized, which are abbreviated as follows: RK_{45}: single-step explicit Runge–Kutta (4, 5) solver [11], $TR - BD$: implicit Runge–Kutta with a trapezoidal rule step and a backward differentiation [20], ABM_{13}: Adams–Bashforth–Moulton multistep solver of variable orders 1 to 13 [15, 47], and $Radau$5 implements the implicit Runge–Kutta method of fifth order introduced by Hairer [18] and implemented for multibody systems [33].

Figure 17 depicts the norm of constraint violations of the system. The solution is accompanied by significant violations with various scales. Obviously, the ABM_{13} method, when utilized judiciously, is more effective than any other method. It may be accepted for short simulation time, but for long simulations, the absence of error control on constraints causes a progressive increase in constraint violations, and consequently, the constraints begin to be violated as a result of the integration process. In addition, the ABM_{13} method is not self-starting and requires the help of a single step scheme to initiate the integration process, which is another error-causing reason among others [15]. On the other hand, $TR - BD$ produces unbounded violations at early simulation time, and $Radau$5 demands excessive computation time in comparison to ABM_{13} (145%) and RK_{45} (230%).

Toward the paper's goal, minimizing constraint violations with explicit solver with shortest computation time, the constraint violations using RK_{45} solver with the Baumgarte stabilization method are illustrated in Figs. 18, 19. Three sets of (α, β) are used as follows:

Fig. 17 Constraint violations without stabilization - Position level - $\delta t = 1$ ms
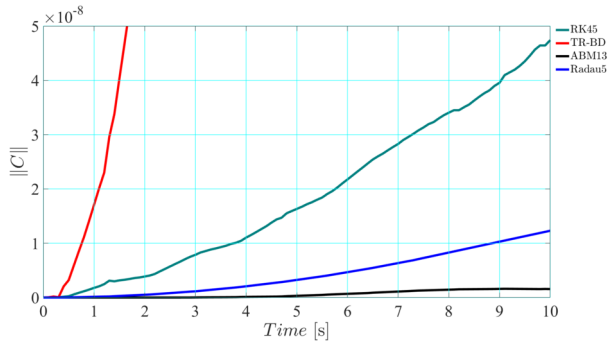

Fig. 18 Constraint violations with Baumgarte stabilization - Position level
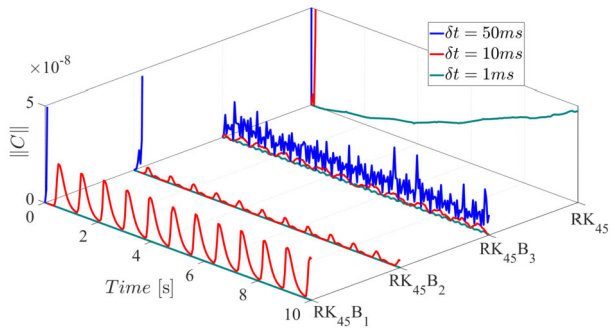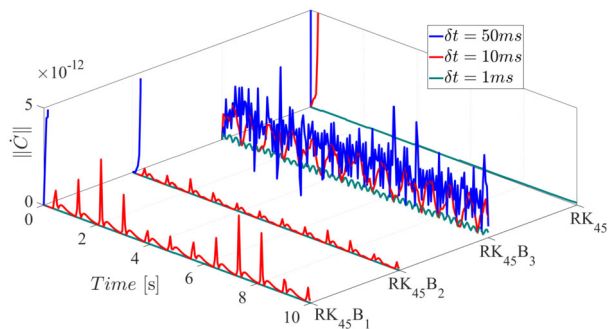

Fig. 19 Constraint violations with Baumgarte stabilization - Velocity level

$RK_{45}B_1 : (\alpha = 5, \beta = 5)$, $RK_{45}B_2 : (\alpha = 5, \beta = 12)$, and $RK_{45}B_3 : (\alpha = 1/\delta t, \beta = \sqrt{2}/\delta t)$, where δt is the time step of the integration. The integration is carried out using different values of δt with $\delta t = 1, 10, 50$ [ms]. The figures show the 1-norms of the constraint vectors, which are the sum of the absolute values of the vector elements.

Obviously, only $RK_{45}B_3$ gives a stable behavior at the position and velocity levels with spaced step time $\delta t = 50$ [ms]. Constant values of (α, β) give unstable behavior for the same step time. Clearly, identifying these values (α, β) is the most difficult aspect of Baumgarte stabilization. For dense time steps, all constant sets of the Baumgarte stabilization method provide a stable behavior. However, there is a significant difference in the calculation time, which we shall discuss in more detail with the proposed method in the paper. It should be noted that $RK_{45}B_3$ depends on the step time, and that utilizing dense steps results in

Fig. 20 Constraint violations
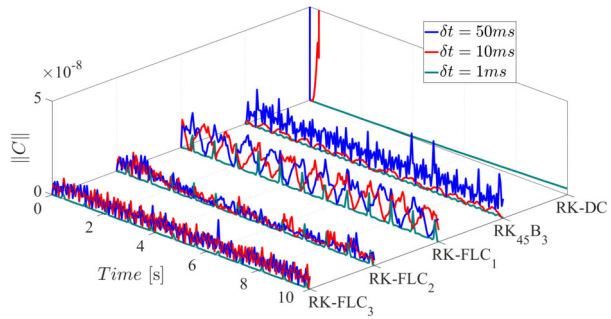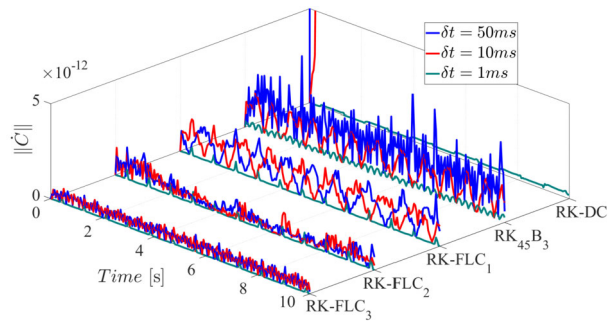with FLC stabilization - Position
level

Fig. 21 Constraint violations
with FLC stabilization - Velocity
level

significant enlargement of the stabilization constants, which may affect the calculation time accordingly.

So, the challenge of introducing the FLC stabilization is to achieve a higher level of performance than the $RK_{45}B_3$ and direct correction approaches by using the explicit integration method, i.e., RK_{45}, with the largest possible step time of integration, i.e., $\delta t = 50$ [ms], and shortest possible calculation time.

In this section, we examine the FLC constraint stabilization method as established in Fig. 6 with the three sets of membership functions shown in Figs. 7–9 for the two inputs, and singleton MF for the output, Fig. 11. These three controllers denoted as $RK - FLC_1$: $(SetI)$, $RK - FLC_2$: $(SetII)$, and $RK - FLC_3$: $(SetIII)$ are using triangular, trapezoidal, and Gaussian MFs respectively. In addition, the FLC has three scaling factors, see Fig. 3, which are held constant throughout the numerical integration process. In this investigation, the values of these constants rely on the permissible error of the constraint functions at position and velocity levels, such that $K_p = 1 \times 10^2$, $K_d = 1 \times 10^{-2}$, and $K_u = 1 \times 10^4$.

Figures 20, 21 show the 1-norms of the constraint violations obtained using the direct correction, Baumgarte, and FLC stabilization methods: $RK_{45} - DC$, $RK_{45}B_3$, and $RK - FLC_{1,2,3}$, respectively, with different values of δt.

It is clear from the two figures that the $RK - DC$ method is ineffective for lengthy simulations as the error accumulates with the simulation time at the position and velocity levels, and that it is only applicable for short simulation time with utilizing extremely intense time steps. As mentioned above, the $RK_{45}B_3$ method gives stable results and with an acceptable violations that do not affect the results of numerical integration. The most important point is the stability of the constraints with using of spaced time steps, here $\delta t = 50$ [ms]. In comparison with these results, the contribution provided by stabilizing the constraints using fuzzy control is noticeable. The three cases, $RK - FLC_{1,2,3}$, show a stable behavior with a lower

Fig. 22 Ninth constraint
violations with FLC
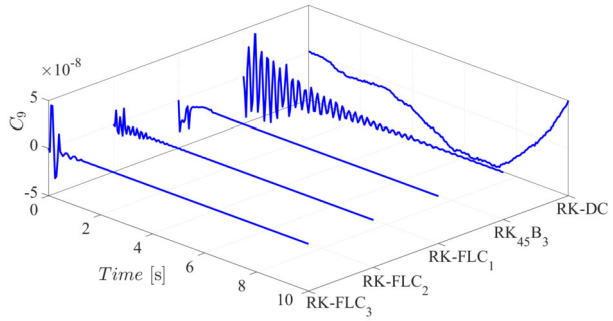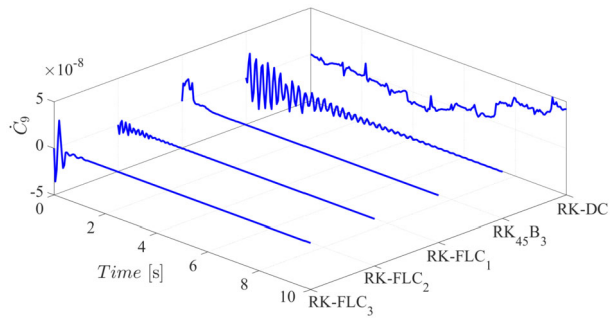stabilization - Position level

Fig. 23 Ninth constraint
violations with FLC
stabilization - Velocity level

level of violations than $RK_{45}B_3$, particularly at the velocity level. It can be concluded that $RK - FLC_3$ with the Gaussian membership function produces better outcomes in terms of stability and violations.

Additionally, all rotational coordinates of the system are included in the last two equations of Eq. (25), i.e., the fourth and fifth equations in Eq. (24), which restrict the relative rotation between the arm and the pendulum. Figures 22, 23 depict the dynamics of the ninth constraint at the position and velocity levels, respectively. It should be noticed that RK-DC remains stable for less than 10% of the entire simulation time, while the $RK_{45}B_3$ shows a highly oscillatory nature for more than 50% of the simulation time before becoming stable. On the other hand, fuzzy methods are distinguished by rapid damping of oscillation, whereas employment of Gaussian functions, $RK - FLC_3$, yields the best possible performance. This is due to the continuity of membership functions of set III, see Fig. 9, while the discontinuous shapes of $RK - FLC_{1,2}$ methods contribute to the oscillating nature of the constraint dynamics.

Concerning the last aspect, the solver's calculation time, which is a key factor when comparing the various approaches, the calculation times for all investigated methods are monitored and presented in Fig. 24 with varying δt values. With $RK - FLC3$, the minimal computation time may be attained. Although the $RK_{45}B_3$ exhibits a near value of calculation time compared to $RK - FLC_3$, the violation is high, see Figs. 20, 21. In addition, the calculation time of $RK_{45}B_3$ grows sharply when dense steps are implemented, going beyond the axe limits. For the same step, it is generally observed that the calculation times for FLC-stabilization methods are often close to each other. Based on the previous explanation, the $RK - FLC_3$ with Gaussian membership function produces better outcomes in terms of stability, violations, and calculation time.
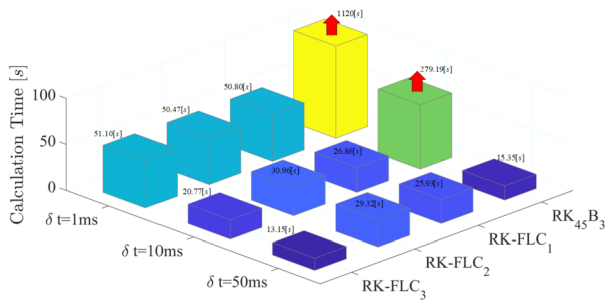
Fig. 24 Calculation time of solvers
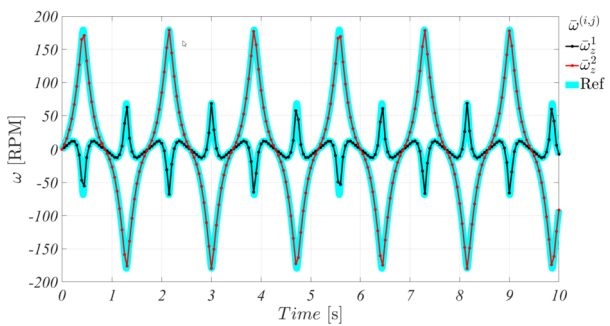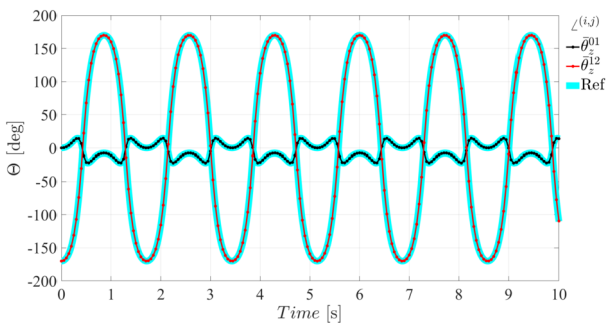

Fig. 25 Comparison with ABM_{13}

Fig. 26 Comparison with ABM_{13} as reference solution

It is worth mentioning that all the stabilization methods described above were performed on the same CPU processor (Intel(R) Core(TM) $i9 - 7920X$, 2.90 GHz), with only one core utilization, the multicore and parallel computation are not employed in this study.

The simulated outputs of local angular velocities and displacement are shown in Figs. 25, 26, respectively. The comparison is carried out between the results of ABM_{13} integrator without stabilization but with dense steps ($\delta t = 0.1$ [ms]) and referred in the figures as the reference curve, and the proposed FLC-stabilization method $RK - FLC_3$ with ($\delta t = 50$ [ms]). It can be concluded, with these results, that the fuzzy approach for constraint stabilization of holonomic multibody systems with rigid bodies and linear constraints at the velocity level is quite successful.

5 Example II: multibody system with nonholonomic constraints

An epicyclic gear train is demonstrated in this section to investigate multibody systems subjected to position and velocity constraints and to examine the proposed method of constraint stabilization utilizing FLC.

In multibody systems, both holonomic and nonholonomic restrictions may coexist. Nonholonomic constraints impose restrictions on generalized velocities, consequently restricting velocities and accelerations. They do not, however, restrict the generalized coordinates. This implies that there are more independent coordinates than independent velocities or accelerations [21, 27, 39]. The transmission of motion through gear trains, using simplistic assumptions, could be considered an example of these nonholonomic constraints-subjugated systems. These assumptions include that the gear-set model is represented by two pitch circles rolling on each other without slipping; gears in contact are considered rigid, the flexibility of gear teeth has been ignored, the backlash between gears is not taken into account, and spacing errors and misalignments are not considered. In this case, the tangential velocity between gear i and j at the contact point is equal, i.e., $\bar{\omega}_z^i \cdot r^i = \bar{\omega}_z^j \cdot r^j$, where r is the radius of the gear and $\bar{\omega}_z$ is the local angular velocity of the gear. It should be noted that, in the case of using Euler angles as orientational coordinates, i.e., $\boldsymbol{\theta}^i = \begin{bmatrix} \phi^i & \theta^i & \psi^i \end{bmatrix}^T$, the local angular velocity can be expressed as $\bar{\omega}_z^i = 2\cos\left(\theta^i\right)\dot{\phi}^i + \dot{\psi}^i$, whereas using Euler parameters, i.e., $\boldsymbol{\theta} = \begin{bmatrix} e_0 & e_1 & e_2 & e_3 \end{bmatrix}^T$, leads to $\overline{\omega}_z^i = -2e_3\dot{e}_0 + 2e_2\dot{e}_1 - 2e_1\dot{e}_2 + 2e_0\dot{e}_3$. In both cases, the constraint equation of equal tangential velocity between the two gears cannot be integrated in time to form a position constraint (nonintegrable).

In the current example, the multibody system of the epicyclic gear train can be assembled using seven bodies with the arrangement listed in Table 3. Unlike the previous example, Euler parameters were utilized to designate the body orientation instead of Euler angles to overcome the problem of kinematic singularities. Thus, a set of seven coordinates will be utilized to define each body's position and orientation, i.e., $\mathbf{q}^i = \begin{bmatrix} \mathbf{R}^{i^T} & \boldsymbol{\theta}^{i^T} \end{bmatrix}^T$, $i = 1, \ldots, 7$, where $\boldsymbol{\theta}^i = \begin{bmatrix} e_0^i & e_1^i & e_2^i & e_3^i \end{bmatrix}^T$ are the Euler parameters of body i. The ring gear is held in place by a rigid joint with the ground. The gear train, see Fig. 27, has two pins that are rigidly attached to the arm body from one end and carry two planet gears from the other end. Accordingly, the total number of generalized coordinates is 49, and the interdependence between them is established by specifying the constraints in accordance with the types of joints as presented in the layout of Table 4. In this table, the number of constraints includes the Euler parameter constraints of each body that have unit magnitude, i.e., $\boldsymbol{\theta}^i \boldsymbol{\theta}^{i^T} = e_0^2 + e_1^2 + e_2^2 + e_3^2 = 1$. In addition to holonomic constraints, nonholonomic constraints are imposed because the engaged gears have zero relative velocities at their contact points. Therefore, the following equations apply:

$$T_2\left(\bar{\omega}_z^2 - \bar{\omega}_z^3\right) + T_5\left(\bar{\omega}_z^5 - \bar{\omega}_z^3\right) = 0, \tag{26}$$

$$(T_2 + 2T_5)\,\bar{\omega}_z^1 - (T_2 + T_5)\,\bar{\omega}_z^3 - T_5\bar{\omega}_z^5 = 0, \tag{27}$$

$$T_2\left(\bar{\omega}_z^2 - \bar{\omega}_z^3\right) + T_7\left(\bar{\omega}_z^7 - \bar{\omega}_z^3\right) = 0, \tag{28}$$

where T_i is the number of teeth of respective gear body and $\bar{\omega}_z^i$ is the angular velocity about its local z-axis. The last constraint equation relates the rotational speed of the arm, planet, and sun gears. Thus, the total number holonomic and nonholonomic constraints becomes 48, and consequently, the system has only one degree of freedom.

Fig. 27 Multibody model epicyclic gear train
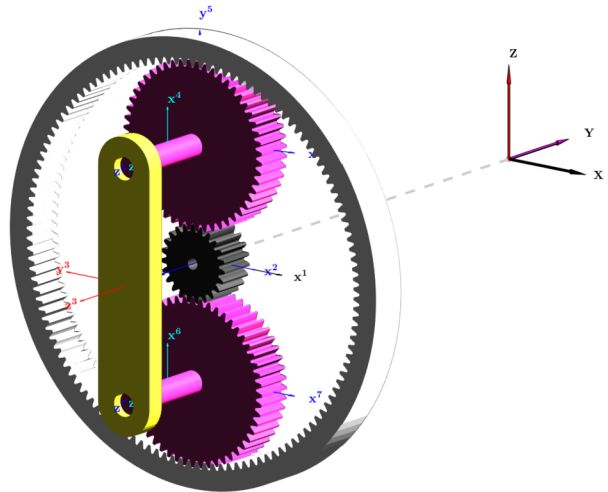
The system is presented by Nada [31], and the simulation results of applying a torque of 0.1 [N.m] on the arm along with the rotational axis (local z^3-axis) are carried out. The direct correction method of constraint stabilization is examined in comparison with the selective coordinates partitioning method, in which the accumulative violations are not acceptable [31].

The nonholonomic constraints, Eqs. (26)–(28) are linear in the generalized velocities, therefore, these equations can be written as

$$\dot{\mathbf{C}}^{nh}(\mathbf{q}, \dot{\mathbf{q}}, t) = \mathbf{G}(\mathbf{q}, t)\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}, t) = \mathbf{0} \tag{29}$$

such that $\mathbf{G}$ and $\mathbf{g}$ are respectively a matrix and a vector that may depend on the system coordinates and time. Thus, the total constraints, in the velocity level, can be constructed by combining the holonomic part, Eq. (2), and the nonholonomic part of Eq. (29) as follows:

$$\dot{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}}, t) = \begin{bmatrix} \dot{\mathbf{C}}^h(\mathbf{q}, t) \\ \dot{\mathbf{C}}^{nh}(\mathbf{q}, \dot{\mathbf{q}}, t) \end{bmatrix} = \begin{bmatrix} \mathbf{C}_q^h\dot{\mathbf{q}} + \mathbf{C}_t^h \\ \mathbf{G}(\mathbf{q}, t)\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}, t) \end{bmatrix} = \mathbf{0} \tag{30}$$

$$\Updownarrow$$

$$\begin{bmatrix} \mathbf{C}_q^h \\ \mathbf{G}(\mathbf{q}, t) \end{bmatrix} \dot{\mathbf{q}} + \begin{bmatrix} \mathbf{C}_t^h \\ \mathbf{g}(\mathbf{q}, t) \end{bmatrix} = \mathbf{0}. \tag{31}$$

The last equation in Eq. (31) represents a direct restriction on the vector of generalized velocities. Consequently, the acceleration constraint equations may be derived by performing a second differentiation of Eq. (30) with respect to time as follows:

$$\ddot{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}}, t) = \begin{bmatrix} \ddot{\mathbf{C}}^h(\mathbf{q}, \dot{\mathbf{q}}, t) \\ \ddot{\mathbf{C}}^{nh}(\mathbf{q}, \dot{\mathbf{q}}, t) \end{bmatrix} = \begin{bmatrix} \mathbf{C}_q^h\ddot{\mathbf{q}} + \left(\mathbf{C}_q^h\dot{\mathbf{q}}\right)_q \dot{\mathbf{q}} + 2\mathbf{C}_{qt}^h\dot{\mathbf{q}} + \mathbf{C}_{tt}^h \\ \mathbf{G}\ddot{\mathbf{q}} + (\mathbf{G}\dot{\mathbf{q}})_q\dot{\mathbf{q}} + \left(\mathbf{g}_q + \mathbf{G}_t\right)\dot{\mathbf{q}} + \mathbf{g}_t \end{bmatrix} = \mathbf{0}.$$
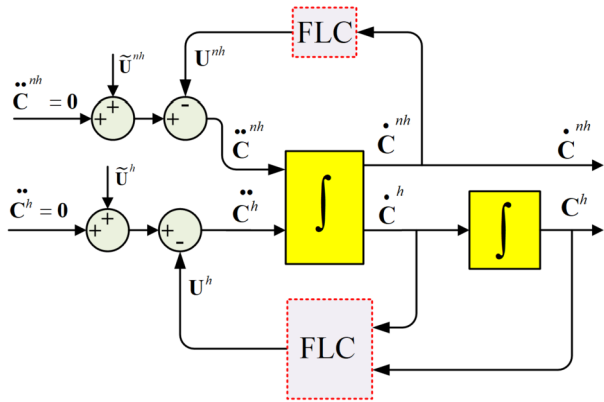
Hence, updating Eq. (6), the generic equations of motions for systems include nonholonomic constraints and can be written as

$$
\begin{bmatrix} \mathbf{M} & \mathbf{C}_q^{hT} & \dot{\mathbf{C}}_{\dot{\mathbf{q}}}^{nhT} \\ \mathbf{C}_q^h & \mathbf{0} & \mathbf{0} \\ \dot{\mathbf{C}}_{\dot{\mathbf{q}}}^{nh} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}} \\ \boldsymbol{\lambda}^h \\ \boldsymbol{\lambda}^{nh} \end{bmatrix} = \begin{bmatrix} \mathbf{Q} \\ \mathbf{Q}_d^h \\ \mathbf{Q}_d^{nh} \end{bmatrix},
\tag{32}
$$

where $\dot{\mathbf{C}}_{\dot{\mathbf{q}}}^{nh} = \mathbf{G}(\mathbf{q}, t)$ is the Jacobian matrix of the nonholonomic constraints, $\mathbf{Q}_d^{nh} = -(\mathbf{G}\dot{\mathbf{q}})_{\mathbf{q}}\dot{\mathbf{q}} - (\mathbf{g_q} + \mathbf{G}_t)\dot{\mathbf{q}} - \mathbf{g}_t$, and $\boldsymbol{\lambda}^{nh}$ are the Lagrange multipliers associated with the nonholonomic constraints. Note that the tangential forces between the contacted gears can be estimated as $\mathbf{F}_T = \dot{\mathbf{C}}_{\dot{\mathbf{q}}}^{nhT} \boldsymbol{\lambda}^{nh}$.

In this case, beside stabilizing the holonomic constraints as demonstrated in Sect. 3, we use FLC with two inputs $(\mathbf{C}^h(\mathbf{q}, t), \dot{\mathbf{C}}^h(\mathbf{q}, t))$ and one output $\mathbf{U}^h$ that compensate the integration errors at two levels. Nonholonomic constraints will need the addition of a further FL controller to stabilize the nonholonomic constraints at the velocity level. Figure 28 shows the modified layout for constraint stabilization of both types, and for the nonholonomic part, it shows only one input and one output. Therefore the governing rules are reduced to five rules only, specifically, the third row of Table 1 since the nonholonomic constraints are $(- \Downarrow)$ Negative Big, $(- \downarrow)$ Negative Small, $(\longleftrightarrow)$ Zero, $(+ \uparrow)$ Positive Small, and $(+ \Uparrow)$ Positive Big, according to permissible margin of error. We can refer to single input single output FLC as $FLC - SISO$ or by the number of utilized rules, as $FLC_{1 \times 5}$.

The $RK - FLC_3$ is employed for stabilizing the holonomic constraints with the same scaling factors, while the nonholonomic constraints are stabilized using $FLC - SISO$ with Gaussian MFs for $\dot{\mathbf{C}}^{nh}(\mathbf{q}, \dot{\mathbf{q}}, t)$ as the only input and the singleton MFs for the output. The control action can be implemented for the multibody system as

$$
\left. \begin{array}{l} \mathbf{C}_\mathbf{q}^h \ddot{\mathbf{q}} = \mathbf{Q}_d^h - \mathbf{U}^h \\ \mathbf{C}_{\dot{\mathbf{q}}}^{nh} \ddot{\mathbf{q}} = \mathbf{Q}_d^{nh} - \mathbf{U}^{nh} \end{array} \right\}.
\tag{33}
$$

The dynamics of the gear train system is characterized by high rotational motion, and therefore the rotational coordinates, i.e., Euler parameters, change steadily. Figures 29, 30 show the violations of the Euler parameter constraints, $\boldsymbol{\theta}^i \boldsymbol{\theta}^{iT} = 1$, $i = 1 \longrightarrow 7$, at the position and velocity levels, respectively. The result confirms the prior case, presented in Example I, where dampening of the constraint dynamics requires approximately 12% of the

Fig. 29 1-norms of constraint violation of the epicyclic gear train
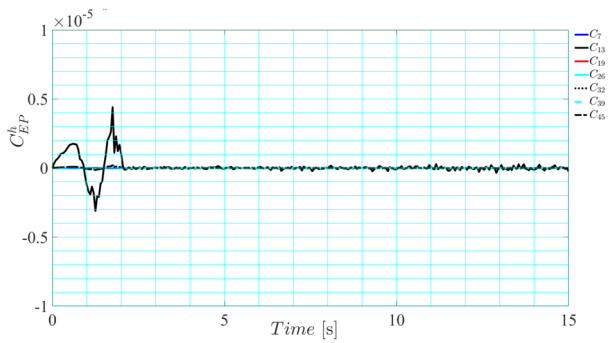
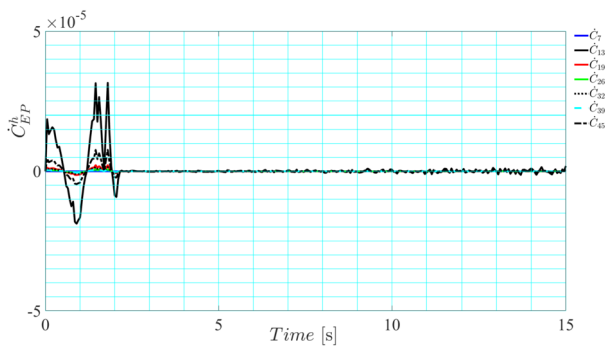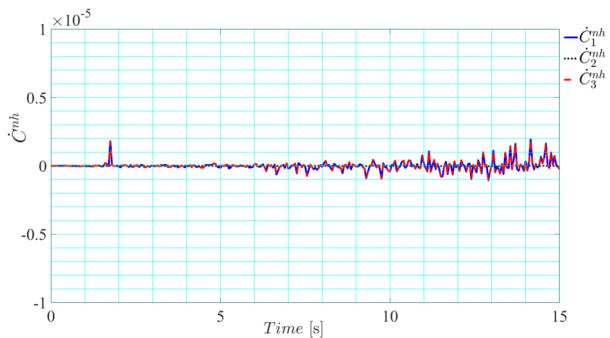Fig. 30 Nonholonomic constraint violation of the epicyclic gear train

Fig. 31 Nonholonomic constraint violation of the epicyclic gear train

simulation time. Moreover, the nonholonomic constraints show a stable dynamic behavior within the range of permissible error, see Fig. 31. It should be mentioned that the dynamic behavior of the nonholonomic constraints under high rotation of the system's components and the exposure to singular configurations throughout this rotation demonstrates the effectiveness of the FLC-stabilization.

Moreover, the norm of the holonomic as well as the nonholonomic constraints is presented in Fig. 32, which exhibits a limited and acceptable error for both types.

Fig. 32 1-norms of constraint violation of the epicyclic gear train
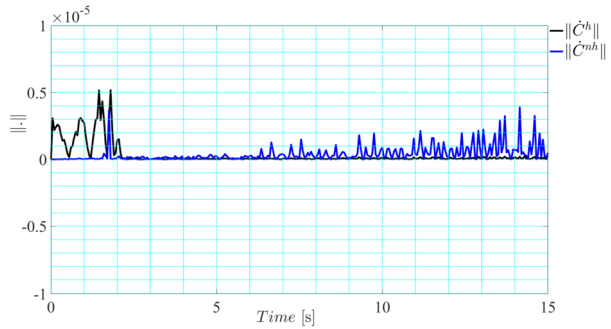
6 Summary and conclusions

Real-time simulation of large multibody systems with time-consuming iterations of the numerical solvers must be avoided. To do this, explicit schemes with deterministic timing behavior are required to be implemented. However, some constraints drifting at the position and velocity level may take place and encounter integration errors. Numerous publications demonstrate constraint stabilization methods for holonomic constraints, whereas only a few address the nonholonomic situation. It is found in the literature that the direct correction method exhibits numerical instabilities and excessive computational time especially in singular configurations. On the other hand, the Baumgarte stabilization approach suffers from the lack of a criterion in the choice of its control parameters. The challenge of introducing the FLC stabilization is to achieve a higher level of numerical stability than the direct correction and Baumgarte approaches by using Runge–Kutta explicit integration with the largest possible step time of integration and the shortest possible computation time. The paper suggests implementing Sugeno-type systems to estimate the output of the Fuzzy controller. For holonomic constraints, the paper proposes an FLC with various sets of membership functions for the input variables, which are the constraint function and its time derivative, and a singleton membership function for the output variable. The output of the controller compensates the constraints drifting due to the numerical integration. For nonholonomic constraints, the FLC is employed with one input variable that is $\mathbf{C}^{nh}$ only. The paper outlined a step by step solution method and explained the method of selecting the FLC scaling constants. The notable enhancement of the FLC-constraint stabilization is to provide a mechanism for regulating the permissible boundaries of the constraint violations, a feature that was missing in other approaches to stabilizing the constraints. The comparisons are carried out using various demonstrative examples of planar and spatial multibody systems subjected to holonomic and nonholonomic constraints, where the simulation is performed on the same CPU processor. As demonstrated by the examples provided in this study, incorporating the FLC within the constraint stabilization implies robust performance against the type of motion (periodic or chaotic), configuration status (determinate or singular), and highly rotated systems subjected to nonholonomic constraints. Based on the comparisons between results with the direct correction, Baumgarte stabilization, and FLC-stabilization with different membership functions, it is found that the FLC with Gaussian membership function produces better outcomes in terms of stability, violations, and computation time and for both types of the system constraints. It can be concluded that the constraint stabilization based on FLC introduces an excellent solution for generic system configuration suitable for lengthy simulations with minimal computation time, which are important factors in appraising the solution technique. The future work includes investigating the FLC-constraint stabilization of multibody

systems with rolling and redundant constraints, flexible multibody, and joints compliance. Future development of the FLC-stabilization will focus on the optimal selection of fuzzy membership functions, and it is anticipated that irregularly distributed functions will yield more effective outcomes.

Appendix 1: List of symbols

l	Length	T	Number of teeth
m	Body mass	$\mathbf{r}$	Global position
g	Gravitational constant	$\mathbf{q}$	Generalized coordinates vector
t	Time	$\bar{\mathbf{u}}$	Local position
δt	Time step of integration	$\mathbf{A}$	Transformation matrix
α	Baumgarte parameter	$\mathbf{C}$	Constraint function
β	Baumgarte parameter	$\mathbf{C_q}$	Jacobian matrix
c^i	Value of fuzzy rule i	$\mathbf{C_q^\dagger}$	Pseudo-inverse of matrix $\mathbf{C_q}$
w^i	Strength of fuzzy rule i	$\overline{\mathbf{G}}$	Velocity transformation matrix
$\boldsymbol{\lambda}$	Lagrange multipliers vector	$\mathbf{I}$	Identity matrix
$\boldsymbol{\theta}$	Orientational parameters	$\mathbf{M}$	Mass matrix
$\overline{\boldsymbol{\omega}}$	Local angular velocity vector	$\mathbf{R}$	Global position of the body frame
Δ	Differential change	$\mathbf{U}$	FLC control output
a, b, c, d	Membership parameter	$\tilde{\mathbf{U}}$	Constraint violations
e_0, e_1, e_2, e_3	Euler parameters	$\mathbf{Q}$	Generalized force vector
A	Probability of input	$\mathbf{Q}_{ex}$	External force vector
K	Scaling factor	$\mathbf{Q}_g$	Gravitational forces
N	Number of fuzzy rules	$\mathbf{Q}_v$	Coriolis inertia forces

Note that the list of symbols and values of Example II is listed in Appendix 2.

Appendix 2: System parameters

Table 2   System parameters of spatial pendulum

Sym.	Description	Value	Sym.	Description	Value
r_a	Width of L-shape arm		J_a	Moment of inertia of	
r_o	Offset of pendulum body	2.54	M_p	Mass of	2.54
ℓ_a	Height of L-shape arm	1.39	J_p	Moment of inertia of	1.39
ℓ_{p1}	Pendulum arm length	2.54	R_3	Betz Optimization	2.54
ℓ_{p2}	Pendulum mass length	2.54	R_4	Tapered (arbitrary)	2.54

Table 3 Multibody system components of the epicyclic gear train

#.	Body name	Index of coords	Inertia properties [Kg], [Kg.m^2]			
			m	I_{xx}	I_{yy}	I_{zz}
1	*Ring*	$1 \rightarrow 7$	6.97	0.25	0.255	0.507
2	*Sun*	$8 \rightarrow 14$	0.85	0.00064	0.00064	0.00087
3	*Arm*	$15 \rightarrow 21$	5.42	0.0032	0.0569	0.05957
4	*Carr*1	$22 \rightarrow 28$	0.832	0.0016	0.0016	0.00093
5	*Plnt*1	$29 \rightarrow 35$	4.41	0.013	0.013	0.024
6	*Carr*2	$36 \rightarrow 42$	0.832	0.0016	0.0016	0.00093
7	*Plnt*2	$43 \rightarrow 49$	4.41	0.013	0.013	0.024

Table 4 Holonomic constraints of the epicyclic gear train

Joint type	Body(i)	Body(j)	# constraints eqs.
Rigid	*Ground*	*Ring*	7
Revolute	*Ground*	*Sun*	6
Revolute	*Ground*	*Arm*	6
Rigid	*Arm*	*Carrier*1	7
Revolute	*Carrier*1	*Planet*1	6
Rigid	*Arm*	*Carrier*2	7
Revolute	*Carrier*2	*Planet*2	6

Declarations

Competing interests The authors declare no competing interests.

References

1. Agúndez, A., García-Vallejo, D., Freire, E.: Linear stability analysis of nonholonomic multibody systems. Int. J. Mech. Sci. **198**, 106392 (2021). https://doi.org/10.1016/j.ijmecsci.2021.106392. https://linkinghub.elsevier.com/retrieve/pii/S0020740321001272

2. Ambrósio, J.A., Gonçalves, J.P.: Complex flexible multibody systems with application to vehicle dynamics. Multibody Syst. Dyn. **6**(2), 163–182 (2001). https://doi.org/10.1023/A:1017522623008. https://link.springer.com/article/10.1023/A:1017522623008

3. Bae, D.-S., Yang, S.-M.: A Stabilization Method for Kinematic and Kinetic Constraint Equations. Springer, Berlin (1990). https://doi.org/10.1007/978-3-642-76159-1_11. http://link.springer.com/10.1007/978-3-642-76159-1_11

4. Bai, Q., Shehata, M., Nada, A.: Review study of using Euler angles and Euler parameters in multibody modeling of spatial holonomic and non-holonomic systems. Int. J. Dyn. Control **10**, 1707 (2022). https://doi.org/10.1007/s40435-022-00913-9. https://link.springer.com/article/10.1007/s40435-022-00913-9

5. Baumgarte, J.: Stabilization of constraints and integrals of motion in dynamical systems. Comput. Methods Appl. Mech. Eng. **1**, 1 (1972). https://doi.org/10.1016/0045-7825(72)90018-7. https://linkinghub.elsevier.com/retrieve/pii/0045782572900187

6. Blajer, W.: A geometric unification of constrained system dynamics. Multibody Syst. Dyn. **1**, 3–21 (1997). https://doi.org/10.1023/A:1009759106323. https://link.springer.com/article/10.1023/A:1009759106323

7. Blajer, W.: Elimination of constraint violation and accuracy aspects in numerical simulation of multibody systems. Multibody Syst. Dyn. **7**, 265 (2002). https://doi.org/10.1023/A:1015285428885. https://link.springer.com/article/10.1023/A:1015285428885

8. Borri, M., Trainelli, L., Croce, A.: The embedded projection method: a general index reduction procedure for constrained system dynamics. Comput. Methods Appl. Mech. Eng. **195**(50–51), 6974–6992 (2006). https://doi.org/10.1016/j.cma.2005.03.010

9. Cardona, A.: Three-dimensional gears modelling in multibody systems analysis. Int. J. Numer. Methods Eng. **40**, 357–381 (1997). https://doi.org/10.1002/(SICI)1097-0207(19970130)40:2. https://onlinelibrary.wiley.com/terms-and-conditions

10. Chiou, J.C., Yang, J.Y., Wu, S.D.: Stability analysis of Baumgarte constraint stabilization technique in multibody dynamic systems. J. Guid. Control Dyn. **22**(1), 160–162 (1999). https://doi.org/10.2514/2.7618

11. Dormand, J.R., Prince, P.J.: A family of embedded Runge-Kutta formulae. J. Comput. Appl. Math. **6**, 19 (1980). https://doi.org/10.1016/0771-050X(80)90013-3

12. Flores, P.: Concepts and Formulations for Spatial Multibody Dynamics, vol. 4. Springer, Berlin (2015). http://link.springer.com/10.1007/978-3-319-16190-7

13. Flores, P., Lankarani, H.M.: Contact Force Models for Multibody Dynamics, vol. 226. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-30897-5. http://link.springer.com/10.1007/978-3-319-30897-5

14. Flores, P., Pereira, R., Machado, M., Seabra, E.: Investigation on the Baumgarte stabilization method for dynamic analysis of constrained multibody systems. In: Proceedings of EUCOMES 2008 - The 2nd European Conference on Mechanism Science, pp. 305–312 (2009). https://doi.org/10.1007/978-1-4020-8915-2_37. https://link.springer.com/chapter/10.1007/978-1-4020-8915-2_37

15. Flores, P., MacHado, M., Seabra, E., Silva, M.T.D.: A parametric study on the Baumgarte stabilization method for forward dynamics of constrained multibody systems. J. Comput. Nonlinear Dyn. **6**, 1 (2011). https://doi.org/10.1115/1.4002338/465842. https://asmedigitalcollection.asme.org/computationalnonlinear/article/6/1/011019/465842/A-Parametric-Study-on-the-Baumgarte-Stabilization

16. Gear, C.W., Leimkuhler, B., Gupta, G.K.: Automatic integration of Euler-Lagrange equations with constraints. J. Comput. Appl. Math. **12–13**, 77–90 (1985). https://doi.org/10.1016/0377-0427(85)90008-1

17. Gim, G., Nikravesh, P.E.: Joint coordinate method for analysis and design of multibody systems: part 2. System topology. KSME J. **7**(1), 26–34 (1993). https://doi.org/10.1007/BF02953142. https://link.springer.com/article/10.1007/BF02953142

18. Hairer, E., Wanner, G.: Stiff differential equations solved by Radau methods. J. Comput. Appl. Math. **111**, 93 (1999). https://doi.org/10.1016/S0377-0427(99)00134-X

19. Hemingway, E.G., O'Reilly, O.M.: Perspectives on Euler angle singularities, gimbal lock, and the orthogonality of applied forces and applied moments. Multibody Syst. Dyn. **44**, 31–56 (2018). https://doi.org/10.1007/s11044-018-9620-0. http://link.springer.com/10.1007/s11044-018-9620-0

20. Hosea, M.E., Shampine, L.F.: Analysis and implementation of TR-BDF2. Appl. Numer. Math. **20**, 21 (1996). https://doi.org/10.1016/0168-9274(95)00115-8

21. Jarzebowska, E.: On derivation of motion equations for systems with non-holonomic high-order program constraints. Multibody Syst. Dyn. **7**(3), 307–329 (2002). https://doi.org/10.1023/A:1015201213396. https://link.springer.com/article/10.1023/A:1015201213396

22. Joli, P., Séguy, N., Feng, Z.Q.: A modular modeling approach to simulate interactively multibody systems with a Baumgarte/Uzawa formulation. J. Comput. Nonlinear Dyn. **3**, 1 (2008). https://doi.org/10.1115/1.2815331/440270. https://asmedigitalcollection.asme.org/computationalnonlinear/article/3/1/011011/440270/A-Modular-Modeling-Approach-to-Simulate

23. Kikuuwe, R., Brogliato, B.: A new representation of systems with frictional unilateral constraints and its Baumgarte-like relaxation. Multibody Syst. Dyn. (2015). https://doi.org/10.1007/s11044-015-9491-6

24. Kim, H.W., Yoo, W.S.: MBD applications in design. Int. J. Non-Linear Mech. **53**, 55–62 (2013). https://doi.org/10.1016/j.ijnonlinmec.2012.10.008

25. Lin, S.-T., Chen, M.-W.: A PID type constraint stabilization method for numerical integration of multibody systems. J. Comput. Nonlinear Dyn. **6**(4), 044501 (2011). https://doi.org/10.1115/1.4002688

26. Marques, F., Souto, A.P., Flores, P.: On the constraints violation in forward dynamics of multibody systems. Multibody Syst. Dyn. **39**, 385–419 (2017). https://doi.org/10.1007/s11044-016-9530-y. https://link.springer.com/article/10.1007/s11044-016-9530-y

27. Modin, K., Verdier, O.: What makes nonholonomic integrators work? Numer. Math. **145**(2), 405–435 (2020). https://doi.org/10.1007/s00211-020-01126-y. https://link.springer.com/article/10.1007/s00211-020-01126-y

28. Müller, A.: Motion equations in redundant coordinates with application to inverse dynamics of constrained mechanical systems. Nonlinear Dyn. **67**(4), 2527–2541 (2012). https://doi.org/10.1007/s11071-011-0165-5. https://link.springer.com/article/10.1007/s11071-011-0165-5

29. Nada, A.: Flexible robotic manipulators: modeling, simulation and control with experimentation. PhD thesis, Mechanical Design and Production Engineering, Cairo University, Egypt (2007). http://mdp.eng.cu.edu.eg/en/m-sc-and-ph-d-thesis/

30. Nada, A.A.: Simplified procedure of sensitivity-based parameter estimation of multibody systems with experimental validation. IFAC-PapersOnLine **54**, 84–89 (2001). https://doi.org/10.1016/J.IFACOL.2021.10.333

31. Nada, A.A., Bashiri, A.H.: Selective generalized coordinates partitioning method for multibody systems with non-holonomic constraints. In: Volume 6: 13th International Conference on Multibody Systems, Nonlinear Dynamics, and Control, p. V006T10A004 (2017). https://doi.org/10.1115/DETC2017-67476. https://asmedigitalcollection.asme.org/IDETC-CIE/proceedings-abstract/IDETC-CIE2017/58202/V006T10A004/258273

32. Nada, A.A., Bishiri, A.H.: Multibody system design based on reference dynamic characteristics: gyroscopic system paradigm. Mech. Based Des. Struct. Mach. **51**, 3372–3394 (2023). https://doi.org/10.1080/15397734.2021.1923526. https://www.tandfonline.com/doi/full/10.1080/15397734.2021.1923526

33. Nada, A.A., Hussein, B.A., Megahed, S.M., Shabana, A.A.: Use of the floating frame of reference formulation in large deformation analysis: experimental and numerical validation. Proc. Inst. Mech. Eng., Proc., Part K, J. Multi-Body Dyn. **224**(1), 45–58 (2010). https://doi.org/10.1243/14644193JMBD208

34. Neto, M.A., Ambrósio, J.: Stabilization methods for the integration of DAE in the presence of redundant constraints. Multibody Syst. Dyn. **10**(1), 81–105 (2003). https://doi.org/10.1023/A:1024567523268. https://link.springer.com/article/10.1023/A:1024567523268

35. Omar, M.A.: Modeling flexible bodies in multibody systems in joint-coordinates formulation using spatial algebra. Adv. Mech. Eng. **6**, 468986 (2014). https://doi.org/10.1155/2014/468986. https://journals.sagepub.com/doi/full/10.1155/2014/468986

36. Pappalardo, C.M., Lettieri, A., Guida, D.: Stability analysis of rigid multibody mechanical systems with holonomic and nonholonomic constraints. Arch. Appl. Mech. **90**, 1961 (2020). https://doi.org/10.1007/s00419-020-01706-2. https://link.springer.com/article/10.1007/s00419-020-01706-2

37. Passas, P., Natsiavas, S., Paraskevopoulos, E.: Numerical integration of multibody dynamic systems involving nonholonomic equality constraints. Nonlinear Dyn. **105**(2), 1191–1211 (2021). https://doi.org/10.1007/s11071-021-06500-5. https://link.springer.com/article/10.1007/s11071-021-06500-5

38. Ponce-Cruz, P., Molina, A., MacCleery, B.: Fuzzy Logic Type 1 and Type 2 Based on LabVIEW™ FPGA, vol. 334. Springer, Berlin (2016). https://doi.org/10.1007/978-3-319-26656-5

39. Rismantab-Sany, J., Shabana, A.A.: On the numerical solution of differential/algebraic equations of motion of deformable mechanical systems with nonholonomic constraints. Comput. Struct. **33**(4), 1017–1029 (1989). https://doi.org/10.1016/0045-7949(89)90437-9

40. Ryan, R.R.: ADAMS — multibody system analysis software. In: Multibody Systems Handbook, pp. 361–402. Springer, Berlin (1990). https://doi.org/10.1007/978-3-642-50995-7_21. http://link.springer.com/10.1007/978-3-642-50995-7_21

41. Schiehlen, W., Guse, N., Seifried, R.: Multibody dynamics in computational mechanics and engineering applications. Comput. Methods Appl. Mech. Eng. **195**(41–43). 5509–5522 (2006). https://doi.org/10.1016/J.CMA.2005.04.024

42. Schwab, A.L., Meijaard, J.P.: Dynamics of flexible multibody systems with non-holonomic constraints: a finite element approach. Multibody Syst. Dyn. **10**(1), 107–123 (2003). https://doi.org/10.1023/A:1024575707338. https://link.springer.com/article/10.1023/A:1024575707338

43. Shabana, A.A.: Computational Dynamics, 3rd edn. Wiley, Chichester (2009). https://doi.org/10.1002/9780470686850. http://doi.wiley.com/10.1002/9780470686850

44. Shabana, A.A.: Dynamics of Multibody Systems. Cambridge University Press, Cambridge (2013). https://doi.org/10.1017/CBO9781107337213. http://ebooks.cambridge.org/ref/id/CBO9781107337213

45. Shabana, A.A., Hussein, B.A.: A two-loop sparse matrix numerical integration procedure for the solution of differential/algebraic equations: application to multibody systems. J. Sound Vib. **327**(3–5), 557–563 (2009). https://doi.org/10.1016/j.jsv.2009.06.020

46. Shah, S.V., Nandihal, P.V., Saha, S.K.: Recursive dynamics simulator (ReDySim): a multibody dynamics solver. Theor. Appl. Mech. Lett. **2**(6), 063011 (2012). https://doi.org/10.1063/2.1206311

47. Shampine, L.F., Reichelt, M.W.: The Matlab ODE suite. SIAM J. Sci. Comput. **18**(1), 1–22 (1997). https://doi.org/10.1137/S1064827594276424

48. Sherif, K., Nachbagauer, K., Steiner, W.: On the rotational equations of motion in rigid body dynamics when using Euler parameters. Nonlinear Dyn. **81**, 343 (2015). https://doi.org/10.1007/s11071-015-1995-3. http://link.springer.com/10.1007/s11071-015-1995-3

49. Siddique, N.H., Adeli, H.: Computational Intelligence: Synergies of Fuzzy Logic, Neural Networks, and Evolutionary Computing. Wiley, New York (2013)

50. Valasek, M., Sika, Z., Vaculin, O.: Multibody formalism for real-time application using natural coordinates and modified state space. Multibody Syst. Dyn. **17**(2–3), 209–227 (2007). https://doi.org/10.1007/s11044-007-9042-x. https://link.springer.com/article/10.1007/s11044-007-9042-x

51. Wang, X., Haug, E.J., Pan, W.: Implicit numerical integration for design sensitivity analysis of rigid multibody systems. Mech. Based Des. Struct. Mach. **33**(1), 1–30 (2005).

52. Wehage, R.A., Haug, E.J.: Generalized coordinate partitioning for dimension reduction in analysis of constrained dynamic systems. J. Mech. Des. **104**(1), 247–255 (1982). https://doi.org/10.1115/1.3256318

53. Wehage, K.T., Wehage, R.A., Ravani, B.: Generalized coordinate partitioning for complex mechanisms based on kinematic substructuring. Mech. Mach. Theory **92**, 464–483 (2015). https://doi.org/10.1016/j.mechmachtheory.2015.06.006

54. Zahariev, E., McPhee, J.: Stabilization of constraints of multibody system dynamics. Mech. Based Des. Struct. Mach. **31**, 25–55 (2003). https://doi.org/10.1081/SME-120017108. http://www.tandfonline.com/doi/abs/10.1081/SME-120017108

55. Zahariev, E., McPhee, J.: Stabilization of multiple constraints in multibody dynamics using optimization and a pseudo-inverse matrix. Math. Comput. Model. Dyn. Syst. **9**, 417–435 (2003). https://doi.org/10.1076/MCMD.9.4.417.27898

56. Zhang, X.Q., Yang, B., Yang, C., Xu, G.N.: Research on ABS of multi-axle truck based on ADAMS/Car and Matlab/Simulink. Proc. Eng. **37**, 120–124 (2012). https://doi.org/10.1016/J.PROENG.2012.04.213