

Comparison of Two Alternate Methods for Distributed Forward Dynamic Simulation of a Four-Bar Linkage

WASEEM AHMED KHAN

Department of Mechanical Engineering
McGill University
3480 University Street, #421
Montreal, Quebec, Canada, H3A2A7
email: wakhan@cim.mcgill.ca
Web: <http://www.cim.mcgill.ca/~wakhan>

VENKAT KROVI*

Mechanical and Aerospace Engineering
State University of New York at Buffalo
318 Jarvis Hall
Buffalo, NY 14260
email: vkrovi@eng.buffalo.edu
Web: <http://www.eng.buffalo.edu/~vkrovi>

* Contact Author

Abstract: In this paper, we examine the modular development of two alternate methods for distributed computation of the forward dynamics simulations of constrained mechanical systems such as four-bar linkages. We exploit the natural spatial parallelism of closed-chain linkages, initially, for the modular development of overall dynamics and subsequently, for the distributed numerical simulation of the dynamics. Traditionally, the numerical simulation problem of constrained mechanical systems has been treated as two separate stages: the problem of algorithm development and the subsequent numerical problem of advancing the discretized differential equations in time. However the potential numerical instabilities arising from the formulation stiffness of the algorithm development stage has the potential to hinder the subsequent numerical integration stage. These aspects are also explored during the evaluation of the two alternative approaches for the distributed forward dynamics simulation of a four-bar linkage and preliminary results to quantify the *overall* computational efficiency and accuracy are presented.

1 Introduction

In the last quarter century, dynamics simulation tools have seen manifold increases in terms of their usage in the design, analysis, parametric refinement and ultimately model-based control of a variety of multibody systems such as vehicles, heavy machinery, spacecraft and robots. In the absence of efficient, general-purpose, closed-form analytical methods, numerical simulation methods have taken a premier position for simulation of such multibody systems. The interested reader may refer to a number of books on the subject (Ascher and Petzold, 1998; Garcia de Jalón and Bayo, 1994; Haug, 1989; Schiehlen, 1990; Shabana, 1989) for further details on

the wide variety of formulations and computational methods that exist in the literature for numerical implementation of multibody simulations.

While efficient formulations exist for serial chain and tree structured multibody systems, the adaptation of these methods for the simulation of closed-chain linkages and parallel manipulators is relatively more difficult. Such systems possess one or more closed kinematic loops, requiring the introduction of algebraic (typically non-linear) constraints into the formulation. In our work, we will restrict our attention to the forward dynamic simulation of this class of constrained mechanical systems.

At the outset, we note that the configuration of such constrained multibody systems can be described by a variety of sets of coordinates. The suitable selection of a set of *configuration coordinates* is of particular importance due to its impact both on the ease of formulation and the subsequent computational efficiency. While the use of expanded sets of *dependent configuration coordinates* linked together by *constraining relations* is considered more appropriate for general-purpose analysis than the use of a minimal set of independent configuration coordinates, considerable variety and choice in selection of such sets exists. These choices may be broadly classified into: (i) relative; (ii) reference point and (iii) natural (or fully Cartesian), each bringing its corresponding share of advantages and shortcomings which are discussed in detail in Garcia de Jalón and Bayo (1994). In our case, we will focus our attention on the use of sets of *relative coordinates*, parameterizing the relative degrees of freedom at kinematic articulations. By facilitating direct control of joint-based actuators and enabling a minimal description of the configuration of open-chain systems, such sets of relative coordinates have found extensive use in the mechanisms and robotics community despite other

shortcomings in the form of creation of transcendental constraint relations or of relatively increased complexity of formulations of equations of motion.

In this paper, we examine both the development and performance-evaluation of two alternate methods for *modular and distributed forward dynamic simulations of constrained mechanical systems using such relative coordinates*. The emphasis on modular development is to promote reuse of existing components. Specifically, we will consider that the equations of motion of the individual subsystems/modules are well known and available and that an overall system may be composed of several serial or tree structured individual subsystems plus sets of holonomic constraints. In particular, we will examine exploiting the spatial parallelism (McMillan, 1994) that is inherent in closed kinematic chains to pursue a modular composition of the overall system dynamics. Such simulations are typically also computationally expensive and hence effective methods for distribution of computational load on multiple processors (associated with the composing subsystems) is attractive. The traditionally adopted solution approach is to: augment the unconstrained system dynamics with the differentiated constraints and Lagrange multipliers; convert the augmented system into a system of Ordinary Differential Equations (ODEs) by a variety of methods (to be discussed in the next section); which are then numerically integrated with appropriately applied stabilization or regularization techniques (Ascher and Petzold, 1998; Garcia de Jalón and Bayo, 1994). While good performance of such approaches has been reported for unified system solution, in our work we wish to their applicability and viability in the distributed computation domain.

Specifically, we will examine two approaches – a compliance-based method (e.g. Wang *et al.*, 2000) and a projection-based method (Yun and Sarkar, 1998) – from the viewpoint of *distributing the dynamics computation*. Compliance based methods use artificial mechanical compliance elements (virtual springs and dampers) to *approximate* the constraint forces. Such methods are attractive for distribution of the forward dynamics simulation because they permit explicit approximation of loop-closure constraint forces and can then effectively decouple the numerical integration of component dynamical sub-systems. In contrast, projection-based methods, require projection of the overall dynamics equations onto the feasible motion directions (the reduced-dimensional space of independent generalized velocities), which can add to the computation complexity. However, by permitting the inclusion of various stabilization and regularization methods, this latter approach shows considerable promise for ensuring consistency of the constraints over long periods of time in the presence of numerical disturbances.

Thus, in this paper, we wish to compare the compliance based and projection-based approaches focusing specifically on: (i) the modular development of dynamics formulation of an entire closed-loop manipulator by a composition of the dynamics of the component subsystems; (ii) re-distributing

the computation of the forward dynamics simulations back to the individual subsystems; and (iii) quantifying the relative merits in terms of the relative computational efficiency and accuracy of the two approaches. These aspects will be studied in the context of the distributed forward dynamic simulation of a planar four-bar, whose overall dynamic equations are assumed to result from constraining the independent dynamics of a 2 degree-of-freedom (d.o.f.) left chain and a 1 d.o.f. right chain by means of holonomic constraints.

2 Background

One promising method to overcome the complexity of robotic systems consists of breaking down the system into independent subsystems, which can be mapped onto distributed/parallel processing elements, at the algorithmic or natural body levels. Henrich and Höniger (1997) present a brief review and a preliminary taxonomy of the different levels of parallelism that have been explored in the context of robotic applications and note that parallelization at all levels may not be possible. Results obtained by parallelizing algorithms vary depending on the degree of dependency and coupling among the equations. While image processing problems (Chaudhary and Aggarwal, 1990) can be broken down quite well by dividing the image into smaller independent blocks, the problems of dynamic simulation of constrained mechanical systems is a strongly coupled problem and the task is not trivially parallelizable (Fijany and Bejczy, 1992; Zoyama, 1993).

Fijany and Bejczy (1992) survey many of the methods developed for parallelization of dynamics algorithms, both at the algorithmic level and at the natural body level, for serial chain manipulators. Most work has focused on fine grain parallel algorithms for implementation on special purpose computational architectures (Lee and Chang, 1986; Lee and Chang, 1988; Sadayappan *et al.*, 1989). The primary motivation behind such methods is the desire to speed up computation to satisfy the required real-time constraints and not the requirement for modularity. In contrast, McMillan (1994) proposed and evaluated the use of a combination of spatial parallelism, based on the structural parallelism of a multi-arm or multi-legged system, and temporal parallelism, to compute the forward dynamics of individual chains, examining synchronization requirements and load balancing and our proposed distribution approach is more in this vein.

Several efficient algorithmic approaches have been developed for forward dynamics computation of *serial-chain* and *tree-structured* multibody systems. The two principal approaches are: Composite Rigid Body Methods (CRBM) (Walker and Orin, 1982) which have a computational complexity of $O(N^3)$ but are highly effective for typical robot arms; and Articulated Body Methods (ABM) (Featherstone, 1983) with fast recursive efficient algorithms of $O(N)$ complexity for simulation of longer serial-chain and tree structured dynamic systems. Ascher *et al.* (1997) unified the two seemingly disparate methods by showing that the

different dynamics algorithms (ABM, CRBM) can be derived as different elimination methods for solving an augmented system of Differential Algebraic Equations (DAEs). They also highlight the potential numerical instabilities (“formulation stiffness”) that can arise from separating the treatment of the forward dynamics problem of computing the system accelerations from the numerical integration problem of advancing the discretized differential equations in time and advocate a global, unified view.

The dynamics of *mechanical systems with closed loops* can be formulated as a system of ODEs whose solutions are required to satisfy additional holonomic (algebraic) equations resulting from cutting the loops (Featherstone, 1987). The dynamics of mechanical systems with holonomic constraints can be formulated as Lagrangian equations of the first kind (Arnold, 1989), as:

$$\dot{\mathbf{q}} = \mathbf{v} \quad (1)$$

$$\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} = \mathbf{f}(\mathbf{q}, \mathbf{v}, t, \mathbf{u}) - \mathbf{A}(\mathbf{q})^T \boldsymbol{\lambda} \quad (2)$$

$$\mathbf{C}(\mathbf{q}) = \mathbf{0} \quad (3)$$

where

\mathbf{q} is the n -dimensional vector of generalized coordinates.

\mathbf{v} is the n -dimensional vector of generalized velocities.

$\mathbf{M}(\mathbf{q})$ is the $n \times n$ dimensional inertia matrix.

$\mathbf{f}(\mathbf{q}, \mathbf{v}, t, \mathbf{u})$ is the n -dimensional vector of external forces.

\mathbf{u} is the vector of actuator forces/torques.

$\mathbf{C}(\mathbf{q})$ is a m -dimensional vector of holonomic constraints.

$\mathbf{A}(\mathbf{q}) = \frac{\partial \mathbf{C}}{\partial \mathbf{q}}$ is the $m \times n$ constraint Jacobian matrix.

$\boldsymbol{\lambda}$ is the m -dimensional vector of Lagrange multipliers.

The resulting formulations in non-minimal (redundant) sets of coordinates yields an often simpler, albeit larger, system of index-3 Differential Algebraic Equations (DAE). The development of such models for the entire system using augmented Lagrangian based models, initially in a redundant set of coordinates with a subsequent determination of the multipliers is also attractive since the given models can be used for both forward and inverse dynamics.

The solution of a system of index-3 DAEs by direct finite difference discretization is not possible using explicit discretization methods. The two principal approaches adopted for the forward dynamics simulation of such systems are: (i) *Direct elimination of the surplus variables* using the position-level algebraic constraints to explicitly reduce the index-3 DAE to an ODE in a minimal set of generalized coordinates (conversion into Lagrange’s Equations of the 2nd kind). The resulting (smaller size) ODE can then be integrated using ODE methods without worrying about the stability issues. However, such a reduction cannot be done in general, and even when it can, the obtained differential equations are typically complicated (Kecskemethy *et al.*, 1997). (ii) *Converted ODE approach* wherein all the algebraic position and velocity level constraints are differentiated and represented at the acceleration level to obtain an augmented index-1 DAE (in terms of both the

unknown accelerations and the unknown multipliers). Differentiating the position constraints, Eqn. 3, with respect to time, yields the velocity-level constraints:

$$\dot{\mathbf{C}} = \mathbf{A}(\mathbf{q})\mathbf{v} = 0 \quad (4)$$

and a further differentiation with respect to time yields the acceleration level constraints as:

$$\ddot{\mathbf{C}} = \mathbf{A}(\mathbf{q})\dot{\mathbf{v}} + \dot{\mathbf{A}}(\mathbf{q})\mathbf{v} = 0 \quad (5)$$

Thus, Eqn. 2 can then be written together with Eqn. 5 as an index-1 DAE as:

$$\begin{bmatrix} \mathbf{M} & \mathbf{A}^T \\ \mathbf{A} & \mathbf{0} \end{bmatrix}_{(n+m) \times (n+m)} \begin{bmatrix} \dot{\mathbf{v}} \\ \boldsymbol{\lambda} \end{bmatrix}_{(n+m) \times 1} = \begin{bmatrix} \mathbf{f} \\ -\dot{\mathbf{A}}(\mathbf{q})\mathbf{v} \end{bmatrix}_{(n+m) \times 1} \quad (6)$$

Such index-1 DAEs may be solved by: simply eliminating the Lagrange multipliers in favor of the unknown accelerations leaving a system of ODEs (Ascher and Petzold, 1998); explicitly computing the Lagrange multipliers by a projection into the constrained force space (Murray *et al.*, 1994); approximating the Lagrange multipliers using artificial compliance elements such as virtual springs and dampers (Wang *et al.*, 2000;); or projecting the equations of motion onto the tangent space of the constraint manifold in a variety of ways to obtain constraint-reaction free equations of motion (Garcia de Jalón and Bayo, 1994). These aspects will be explicitly discussed in the next section.

Some of the drawbacks of the converted ODE approach include the need to provide additional consistent initial conditions and the fact that the differentiated constraint manifold is mildly unstable resulting in drift of the state from the position constraint manifold. While the growth rate can be reduced by lowering the error tolerance and by using smaller step-sizes or greater numerical precision, this comes at the cost of longer and more expensive computations.

Hence, most constrained multibody methods also combine (one or more) of the following methods for improved numerical solution using explicit discretization methods (Ascher and Petzold, 1998): (a) Coordinate projection of the state of the system onto the constraint manifold at frequent intervals to ensure maintenance of the algebraic constraint; (b) Computing a local velocity-level parameterization and integrating the ODE on the constraint manifold (in the independent coordinates); or (c) Creation of an artificial first or second-order dynamical system which has the algebraic constraint as its attractive equilibrium configuration (Baumgarte, 1983). While Baumgarte’s technique is very popular in the engineering application community, principally due to the resulting augmented ODE formulation, the practical selection of the parameters of the stabilization system depends both on the discretization methods and step-size and is widely regarded as an open research problem (Ascher *et al.*, 1995).

3 Converted ODE Approaches

Index-1 DAE systems resulting from the converted ODE approach are typically solved using one of the following three approaches: (a) direct Lagrange multiplier elimination;

(b) compliance-based; or (c) projection-based.

In the *direct Lagrange multiplier elimination* approaches, a simultaneous solution of the augmented linear system of equations in Eqn. 6 is possible at each time step. While we note that an explicit inversion of the augmented system may be avoided by adopting a Gaussian elimination method, the overall approach may still be denoted as:

$$\begin{bmatrix} \dot{\mathbf{v}} \\ \dot{\boldsymbol{\lambda}} \end{bmatrix} = \begin{bmatrix} \mathbf{M} & \mathbf{A}^T \\ \mathbf{A} & \mathbf{0} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{f} \\ \boldsymbol{\varphi} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1(\mathbf{q}, \mathbf{v}) \\ \mathbf{f}_2(\mathbf{q}, \mathbf{v}) \end{bmatrix} \quad (7)$$

Thus the overall system may now be written as a system of first order ODEs as:

$$\dot{\mathbf{x}}_{2n \times 1} = \begin{bmatrix} \dot{\mathbf{q}}_{n \times 1} \\ \dot{\ddot{\mathbf{q}}}_{n \times 1} \end{bmatrix} = \begin{bmatrix} \mathbf{v} \\ \mathbf{f}_1(\mathbf{q}, \mathbf{v}) \end{bmatrix} \quad (8)$$

which may then be integrated using standard codes. Note that, in principle, the index-reduced system Eqn. 8 needs more initial conditions than the original system Eqn. 1-3 to specify a unique solution. The main advantage is conceptual simplicity and simultaneous determination of the accelerations and the Lagrange multipliers by solving a *linear* system of equations. However, the constraint conditions may be progressively violated, especially in the presence of large step sizes, leading to unacceptably large errors even for short duration simulations.

In the *compliance-based approaches*, the loop-closure constraints are *relaxed* and replaced using virtual springs and dampers. Using such virtual springs can be considered as a form of penalty formulation (see Section 5.1.4 of Garcia de Jalón and Bayo (1994)) which incorporates the constraint equations as a dynamical system penalized by a large factor. The Lagrange multipliers are *approximated* using a force-law (based the extent of the constraint violation and an assumed spring stiffness) and eliminated from the list of $n+m$ unknowns leaving behind a system of $2n$ first order ODEs. While the sole initial drawback may appear to be restricted to the numerical ill-conditioning due to selection of large penalty factors, it is important to note that penalty approaches only *approximate* the true constraint forces and can create unanticipated problems (as will be discussed later).

Finally, the class of *projection-based approaches* seek to take the constraint-reaction containing dynamical equations into the *orthogonal* and *tangent* subspaces of the vector space of the system's generalized velocities. Let $\mathbf{S}(\mathbf{q})$ be a $n \times (n-m)$ dimensional full rank matrix whose column space is in the null space of $\mathbf{A}(\mathbf{q})$ i.e. $\mathbf{A}(\mathbf{q})\mathbf{S}(\mathbf{q}) = \mathbf{0}$. The orthogonal subspace is spanned by the so-called constraint vectors (forming the rows of the matrix $\mathbf{A}(\mathbf{q})$) while the tangent subspace *complements* this orthogonal subspace in the overall generalized velocity vector space. All *feasible* dependent velocities, $\dot{\mathbf{q}}$, of a constrained multibody system necessarily belong to this tangent space, appropriately called the *space of feasible motions*. This space is spanned by the columns of $\mathbf{S}(\mathbf{q})$ and is parameterized by an $n-m$ dimensional vector of independent velocities, $\mathbf{v}(t)$, yielding the expression for the feasible dependent velocities as $\dot{\mathbf{q}} = \mathbf{S}(\mathbf{q}) \mathbf{v}(t)$.

It is very important to note two factors at this stage. *First*, the initial selection of the set of configuration coordinates plays an important role here. In particular, while it is always possible to create a Riemannian configuration space (and consequently vector spaces for the generalized velocities) using sets of relative coordinates, special care needs to be exercised when treating configuration spaces created with other sets of generalized coordinates (such as Cartesian coordinates of the bodies in SE(2) or SE(3)). This is because the generalized velocity space for Cartesian coordinates need not necessarily form a vector space and hence, the notion of orthogonal complement subspaces, which exist only in a Riemannian setting, needs to be examined carefully. For example, in Blajer *et al.* (1994), the local task-space coordinates (consisting only translations) form a Riemannian space permitting the orthogonal decomposition carried out. This is the other motivating factor for restricting ourselves to joint-based relative-coordinate descriptions of the configuration of the system. *Second*, a family of choices exists for the selection of projection between dependent and independent velocities (including the case where the set of independent velocities form a proper subset of the original set of dependent velocities) and each such choice can give rise to a different $\mathbf{S}(\mathbf{q})$. See Garcia de Jalón and Bayo (1994) for a description of the many possibilities as well as the determination of the projection matrices determining the transformations between dependent and independent velocities.

However, once a projection is selected, the dynamic equations of motion can now be projected on to the instantaneous feasible motion directions, to obtain the so-called the so-called constraint-reaction-free equations of motion. Pre-multiplying both sides of Eqn. 2 by \mathbf{S}^T and noting that $\mathbf{S}^T \mathbf{A}^T = \mathbf{0}$ we get:

$$\mathbf{S}^T \mathbf{M}(\mathbf{q}) \dot{\mathbf{v}} = \mathbf{S}^T \mathbf{f}(\mathbf{q}, \mathbf{v}, t, \mathbf{u}) \quad (9)$$

Note that this is a system of $n-m$ 2nd order differential equations in the n dependent accelerations. All further steps for the solution of this system of equations may be performed either in terms of *dependent* or *independent* coordinates, as discussed in Chapter 5 of Garcia de Jalón and Bayo (1994).

For example, the m acceleration-level constraints shown in Eqn. 5 may be appended to this system resulting in a system of $2n$ first-order ODEs in the state vector consisting of the n dependent velocities and n dependent accelerations. Note that while the notion of dependent and independent velocities is not explicitly considered, this is implicit in the selection of \mathbf{S}^T . Alternatively, by explicitly considering the same local projection used to determine the feasible motion directions, the constraint-reaction-free equations of motion may be expressed in terms of the independent coordinates (Serna *et al.*, 1982). The final solution may be obtained either by numerically integrating a system of $2(n-m)$ first-order ODEs in the $(n-m)$ independent velocities and $n-m$ independent accelerations and solving the position problem at each step or by numerically integrating a system of $2n-m$ first-order ODEs in the n dependent velocities and $n-m$

independent accelerations. The benefits accumulate from two sources. *First*, since the feasible motion directions are guaranteed to be tangent to the holonomic constraint manifold, with an adequately small step size the resulting integrated solution is guaranteed to maintain the constraints. *Second*, the integration of reduced-order system of ODEs (with either $n-m$ or $2n-m$ states) also reduces the error that can ‘creep in’ due to numerical integration.

4 The Two Approaches Under Consideration

4.1 Compliance based Method (Method A)

In Wang *et al.* (2000), the Lagrange multiplier λ in Eqn. 2 is explicitly calculated as a restoring force provided by a virtual spring. This restoring force, proportional to the extent of constraint violation, can be expressed as $\lambda_i = K_i C_i(\mathbf{q})$ where K_i is the spring constant and $C_i(\mathbf{q})$ is the constraint violation in the direction of the respective λ_i . By substituting the value of λ in Eqn. 2, the final ODE system can be written as:

$$\begin{aligned} \dot{\mathbf{x}}_{2n \times 1} &= \begin{bmatrix} \dot{\mathbf{q}}_{n \times 1} \\ \ddot{\mathbf{q}}_{n \times 1} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{v} \\ \mathbf{M}^{-1}(\mathbf{f}(\mathbf{q}, \mathbf{v}, t, \mathbf{u}) - \mathbf{A}(\mathbf{q})^T (\mathbf{K}\mathbf{C}(\mathbf{q}))) \end{bmatrix} \end{aligned} \quad (10)$$

where $\mathbf{K} = \text{diag}[K_i]$ and $\mathbf{C}(\mathbf{q})$ is the vector of constraint violations.

4.2 Projection-based Method (Method B)

The derivation process discussed by Sarkar and Yun (1998) very closely resembles the work of Serna *et al.* (1982) but with the addition of Baumgarte stabilization. The formulation in terms of rheonomous constraints permits easy incorporation of Baumgarte stabilization and additionally can be easily specialized for the scleronomous case. The holonomic constraints $\mathbf{C}(\mathbf{q}) = \mathbf{0}$ are approximated by a first order system of the form:

$$\dot{\mathbf{C}}(\mathbf{q}) + \sigma \mathbf{C}(\mathbf{q}) = \mathbf{0}, \quad \sigma > 0 \quad (11)$$

where σ is the rate of convergence. The equilibrium condition for this first order system is the constraint manifold $\mathbf{C}(\mathbf{q}) = \mathbf{0}$ and for any initial condition $\mathbf{q}(0)$, which may not satisfy the holonomic constraint equation $\mathbf{C}(\mathbf{q}(0)) = \mathbf{0}$, the above first order equation guarantees exponential convergence of $\mathbf{C}(\mathbf{q}(t))$ to zero as the time t progresses. The rate of convergence will be determined by σ , which can be chosen based on specific application. By taking $\mathbf{A}(\mathbf{q})$ as the Jacobian of $\mathbf{C}(\mathbf{q})$, Eqn. 11 can be written as:

$$\mathbf{A}(\mathbf{q})\dot{\mathbf{q}} = -\sigma \mathbf{C}(\mathbf{q}) = \mathbf{a}(\mathbf{q}) \quad (12)$$

Then, the general solution of Eqn. 12 is given by:

$$\dot{\mathbf{q}} = \mathbf{v} = \mathbf{S}(\mathbf{q})\mathbf{v}(t) + \boldsymbol{\eta}(\mathbf{q}) \quad (13)$$

where $\mathbf{S}(\mathbf{q})$ is an $n \times (n-m)$ dimensional full rank matrix whose column space is in the null space of $\mathbf{A}(\mathbf{q})$ i.e.

$\mathbf{A}(\mathbf{q})\mathbf{S}(\mathbf{q}) = \mathbf{0}$, $\mathbf{v}(t)$ is an $n-m$ dimensional vector of independent velocities and $\boldsymbol{\eta}(\mathbf{q})$ is the n -dimensional particular solution of Eqn. 11. Differentiating once we get:

$$\dot{\mathbf{v}} = \mathbf{S}(\mathbf{q})\dot{\mathbf{v}}(t) + \dot{\mathbf{S}}(\mathbf{q})\mathbf{v}(t) + \dot{\boldsymbol{\eta}}(\mathbf{q}) = \mathbf{S}(\mathbf{q})\dot{\mathbf{v}}(t) + \boldsymbol{\gamma}(\mathbf{q}, \mathbf{v}) \quad (14)$$

Pre-multiplying both sides of Eqn. 2 by \mathbf{S}^T and noting that $\mathbf{S}^T \mathbf{A}^T = \mathbf{0}$ we get:

$$\mathbf{S}^T \mathbf{M}(\mathbf{q})\dot{\mathbf{v}} = \mathbf{S}^T \mathbf{f}(\mathbf{q}, \mathbf{v}, t, \mathbf{u}) \quad (15)$$

By substituting $\dot{\mathbf{v}}$ from Eqn. 14 into Eqn. 15 and solving for $\dot{\mathbf{v}}$ we get

$$\dot{\mathbf{v}} = -(\mathbf{S}^T \mathbf{M} \mathbf{S})^{-1} (\mathbf{S}^T \mathbf{M} \boldsymbol{\gamma} + \mathbf{S}^T \mathbf{f}(\mathbf{q}, \mathbf{v}, t, \mathbf{u})) \quad (16)$$

The resulting overall system of ODEs may be expressed in state-space form as:

$$\begin{aligned} \dot{\mathbf{x}}_{(2n-m) \times 1} &= \begin{bmatrix} \dot{\mathbf{q}}_{n \times 1} \\ \dot{\mathbf{v}}_{(n-m) \times 1} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{S}\mathbf{v} + \boldsymbol{\eta} \\ -(\mathbf{S}^T \mathbf{M} \mathbf{S})^{-1} (\mathbf{S}^T \mathbf{M} \boldsymbol{\gamma} - \mathbf{S}^T \mathbf{f}(\mathbf{q}, \mathbf{v}, t, \mathbf{u})) \end{bmatrix} \end{aligned} \quad (17)$$

While this method would work for any projection that is independent of the existing constraints, the particular adopted projection (wherein the independent velocities are selected as a proper subset of the original set of dependent velocities) yields a simple and robust formulation.

5 Distributed Forward Dynamics for a Four-bar linkage

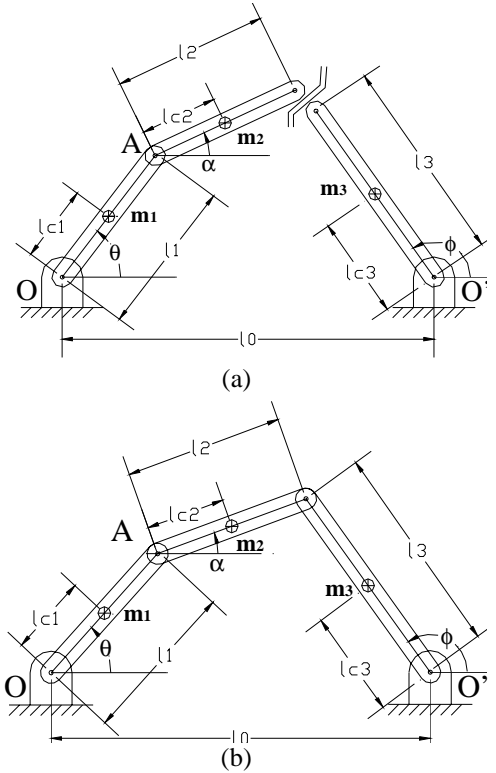


Figure 1. Representation of the four-bar linkage (a) Divided and (b) Composite models.

5.1 Lagrangian Modeling

The modeling process for the four-bar linkage considered here and the selected parameters are the same as in Wang *et al.* (2000) and are summarized here again for clarity. The four-bar linkage (Fig. 1) consists of three movable links (input, coupler link and the output links) of lengths l_1 , l_2 and l_3 respectively whose orientation with respect to the horizontal can be denoted by angles θ , α and ϕ . The mass of each moving link of length l_i is m_i , and the moment of inertia of the moving link about the axis through the center of the mass and perpendicular to the plane of its motion is I_i , where $i=1,2,3$. The mass centers of each link is situated at a distance l_{ci} from the proximal joint of each link.

The equations of motion for the overall system are derived by treating the four-bar as being composed of a left chain (l_1+l_2) and the right chain (l_3), as shown in Fig. 1. The dynamic equations of each chain are derived independently by the Lagrange's method and the equations of the overall system written as an index-3 DAE as:

$$\begin{aligned} \dot{\mathbf{q}} &= \mathbf{v} \\ \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{V}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{G}(\mathbf{q}) &= \mathbf{E}(\mathbf{q})\mathbf{u} - \mathbf{A}^T \boldsymbol{\lambda} \\ \mathbf{C}(\mathbf{q}) &= \mathbf{0} \end{aligned} \quad (18)$$

where:

$$\mathbf{M}(\mathbf{q}) = \begin{bmatrix} m_1 l_{c1}^2 + I_1 + m_2 l_1^2 & m_2 l_1 l_{c2} \cos(\alpha - \theta) & 0 \\ m_2 l_1 l_{c2} \cos(\alpha - \theta) & I_2 + m_1 l_{c2}^2 & 0 \\ 0 & 0 & m_3 l_{c3}^2 + I_3 \end{bmatrix}$$

$$= \begin{bmatrix} [\mathbf{M}_A]_{2 \times 2} & \mathbf{0}_{2 \times 1} \\ \mathbf{0}_{1 \times 2} & [\mathbf{M}_B]_{1 \times 1} \end{bmatrix}$$

$$\mathbf{V}(\mathbf{q}, \dot{\mathbf{q}}) = \begin{bmatrix} -m_2 l_1 l_{c2} \sin(\alpha - \theta) \dot{\alpha}^2 \\ m_2 l_1 l_{c2} \sin(\alpha - \theta) \dot{\theta}^2 \\ 0 \end{bmatrix} = \begin{bmatrix} [\mathbf{V}_A]_{2 \times 1} \\ [\mathbf{V}_B]_{1 \times 1} \end{bmatrix},$$

$$\mathbf{G}(\mathbf{q}) = \begin{bmatrix} m_1 g l_{c1} \cos \theta + m_2 g l_1 \cos \theta \\ m_2 g l_{c2} \cos \alpha \\ m_3 g l_{c3} \cos \phi \end{bmatrix} = \begin{bmatrix} [\mathbf{G}_A]_{2 \times 1} \\ [\mathbf{G}_B]_{1 \times 1} \end{bmatrix},$$

$$\mathbf{q} = \begin{bmatrix} \theta \\ \alpha \\ \phi \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix}, \quad \mathbf{E}(\mathbf{q}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The constraint equations are obtained from the requirement that the left and right chains to stay connected at the cut joint, which can be expressed in matrix form as:

$$\mathbf{C}(\mathbf{q}) = \begin{bmatrix} -l_1 \cos \theta - l_2 \cos \alpha + l_0 + l_3 \cos \phi \\ -l_1 \sin \theta - l_2 \sin \alpha + l_3 \sin \phi \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad (19)$$

5.2 Compliance-based Method (Method A)

By assuming that the state vector $\mathbf{q} = [\mathbf{q}_A^T \quad \mathbf{q}_B^T]^T$ has a state variables belonging to the chain-a and b state variables belonging to the chain-b and that $a + b = n$, the distributed model can be obtained in state-space form as:

$$\begin{aligned} \begin{bmatrix} \dot{\mathbf{x}}_A \\ \ddot{\mathbf{x}}_A \end{bmatrix}_{2 \times 1} &= \begin{bmatrix} \dot{\mathbf{q}}_A \\ \ddot{\mathbf{q}}_A \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{v}_A \\ \mathbf{M}_A^{-1} (\mathbf{E}_A \mathbf{u}_A - \mathbf{V}_A - \mathbf{G}_A - \mathbf{K} \mathbf{A}_A^T \mathbf{C}) \end{bmatrix} \\ \begin{bmatrix} \dot{\mathbf{x}}_B \\ \ddot{\mathbf{x}}_B \end{bmatrix}_{2 \times 1} &= \begin{bmatrix} \dot{\mathbf{q}}_B \\ \ddot{\mathbf{q}}_B \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{v}_B \\ \mathbf{M}_B^{-1} (\mathbf{E}_B \mathbf{u}_B - \mathbf{V}_B - \mathbf{G}_B - \mathbf{K} \mathbf{A}_B^T \mathbf{C}) \end{bmatrix} \end{aligned} \quad (20)$$

where \mathbf{K} is the compliance matrix and \mathbf{C} represents the extent of the constraint violation.

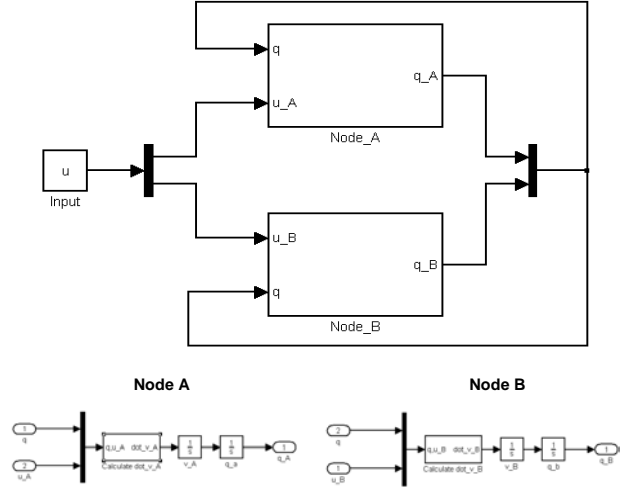


Figure 2: Distributed modeling for a four-bar linkage (Method A)

The two dynamic sub-systems, shown in Eqn. 20, can be simulated in a distributed manner if at every time step: (i) either the information pertaining to $\mathbf{C}(\mathbf{q})$, the extent of the constraint violation, is made available explicitly or (ii) computed by exchanging state information between the two sub-subsystems. This suggests a way to distribute the computational load between two processors, as shown graphically in Fig. 2, wherein each independent sub-part can now be numerically integrated on a different processor. The sole coupling between the two sub-parts is due to the Lagrange multipliers, which are now explicitly calculated using the virtual spring. While this is shown for a “two part system”, the process generalizes easily for “n-part” system.

5.3 Projection-based method (Method B)

Eqn. 17 may be converted into a suitable state-space form as:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{q}} \\ \mathbf{v} \end{bmatrix} = \begin{bmatrix} \mathbf{S}\mathbf{v} + \boldsymbol{\eta} \\ -(\mathbf{S}^T \mathbf{M} \mathbf{S})^{-1} (\mathbf{S}^T \mathbf{M} \boldsymbol{\gamma} + \mathbf{S}^T \mathbf{V} + \mathbf{S}^T \mathbf{G} - \mathbf{S}^T \mathbf{E} \mathbf{u}) \end{bmatrix} \quad (21)$$

By similarly assuming that the state vector \mathbf{q} has a state variables belonging to the chain-A and b state variables

belonging to the chain-b with $a + b = n$ the projected dynamics equations may be partitioned as:

$$\begin{aligned} & \begin{bmatrix} \mathbf{S}_A^T & \mathbf{S}_B^T \\ 0 & \mathbf{M}_B \end{bmatrix} \begin{bmatrix} \mathbf{M}_A & 0 \\ 0 & \mathbf{M}_B \end{bmatrix} \begin{bmatrix} \mathbf{S}_A \\ \mathbf{S}_B \end{bmatrix} \dot{\mathbf{v}} + \begin{bmatrix} \mathbf{S}_A^T & \mathbf{S}_B^T \\ 0 & \mathbf{M}_B \end{bmatrix} \begin{bmatrix} \mathbf{M}_A & 0 \\ 0 & \mathbf{M}_B \end{bmatrix} \begin{bmatrix} \boldsymbol{\gamma}_A \\ \boldsymbol{\gamma}_B \end{bmatrix} \\ & + \begin{bmatrix} \mathbf{S}_A^T & \mathbf{S}_B^T \\ 0 & \mathbf{M}_B \end{bmatrix} \begin{bmatrix} \mathbf{V}_A \\ \mathbf{V}_B \end{bmatrix} + \begin{bmatrix} \mathbf{S}_A^T & \mathbf{S}_B^T \\ 0 & \mathbf{M}_B \end{bmatrix} \begin{bmatrix} \mathbf{G}_A \\ \mathbf{G}_B \end{bmatrix} = \begin{bmatrix} \mathbf{S}_A^T & \mathbf{S}_B^T \\ 0 & \mathbf{E}_B \end{bmatrix} \begin{bmatrix} \mathbf{E}_A & 0 \\ 0 & \mathbf{E}_B \end{bmatrix} \begin{bmatrix} \mathbf{u}_A \\ \mathbf{u}_B \end{bmatrix} \end{aligned} \quad (22)$$

Multiplying we get:

$$\begin{aligned} & [\mathbf{S}_A^T \mathbf{M}_A \mathbf{S}_A + \mathbf{S}_B^T \mathbf{M}_B \mathbf{S}_B] \dot{\mathbf{v}} + [\mathbf{S}_A^T \mathbf{M}_A \boldsymbol{\gamma}_A + \mathbf{S}_B^T \mathbf{M}_B \boldsymbol{\gamma}_B] \\ & + [\mathbf{S}_A^T \mathbf{V}_A + \mathbf{S}_B^T \mathbf{V}_B] + [\mathbf{S}_A^T \mathbf{G}_A + \mathbf{S}_B^T \mathbf{G}_B] \\ & = [\mathbf{S}_A^T \mathbf{E}_A \mathbf{u}_A + \mathbf{S}_B^T \mathbf{E}_B \mathbf{u}_B] \end{aligned} \quad (23)$$

Noting that the first part of Eqn. 23 is $(\mathbf{S}^T \mathbf{M} \mathbf{S})$ and collecting terms we get:

$$\begin{aligned} \dot{\mathbf{v}} &= (\mathbf{S}^T \mathbf{M} \mathbf{S})^{-1} [\mathbf{S}_A^T (\mathbf{E}_A \mathbf{u}_A - \mathbf{M}_A \boldsymbol{\gamma}_A - \mathbf{V}_A - \mathbf{G}_A)] \\ & + (\mathbf{S}^T \mathbf{M} \mathbf{S})^{-1} [\mathbf{S}_B^T (\mathbf{E}_B \mathbf{u}_B - \mathbf{M}_B \boldsymbol{\gamma}_B - \mathbf{V}_B - \mathbf{G}_B)] \end{aligned} \quad (24)$$

Thus, it is now possible to calculate the state vectors forming Eqn. 21 separately as:

$$\begin{aligned} \dot{\mathbf{x}}_A &= \begin{bmatrix} \dot{\mathbf{q}}_A \\ \dot{\mathbf{v}}_A \end{bmatrix}_{(a+n-m) \times 1} \\ &= \begin{bmatrix} \mathbf{S}_A \mathbf{v} + \boldsymbol{\eta}_A \\ -(\mathbf{S}^T \mathbf{M} \mathbf{S})^{-1} (\mathbf{S}_A^T (\mathbf{M}_A \boldsymbol{\gamma}_A + \mathbf{V}_A + \mathbf{G}_A - \mathbf{E}_A \mathbf{u}_A)) \end{bmatrix} \\ \dot{\mathbf{x}}_B &= \begin{bmatrix} \dot{\mathbf{q}}_B \\ \dot{\mathbf{v}}_B \end{bmatrix}_{(b+n-m) \times 1} \\ &= \begin{bmatrix} \mathbf{S}_B \mathbf{v} + \boldsymbol{\eta}_B \\ -(\mathbf{S}^T \mathbf{M} \mathbf{S})^{-1} (\mathbf{S}_B^T (\mathbf{M}_B \boldsymbol{\gamma}_B + \mathbf{V}_B + \mathbf{G}_B - \mathbf{E}_B \mathbf{u}_B)) \end{bmatrix} \end{aligned} \quad (25)$$

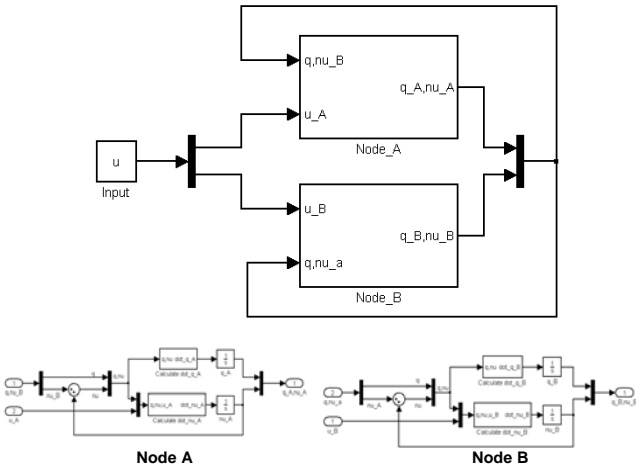


Figure 3: Distributed modeling of a four-bar linkage (Method B)

By examining Eqn. 25, we note that the overall system can be evaluated in a distributed manner if states \mathbf{q}_i and \mathbf{v}_i ($i = a, b$) are made available. This suggests a way to split the calculation of the dynamic equations, as depicted in Fig. 3. Each independent sub-part can now be numerically integrated

on a different processor thereby permitting the distribution of the load. At each time-instant, the complete state of the system needs to be exchanged between the subparts. The coupling between the various sub-parts is due to the existence of the $(\mathbf{S}^T \mathbf{M} \mathbf{S})^{-1}$. In the arrangement shown in Fig. 3, this matrix inverse needs to be computed on each and every processor (although we note that the explicit calculation of the inverse is typically avoided by using an optimal equation solver). Alternatively, state information from the slave processors could be collected by a master processor at each time instant, the $(\mathbf{S}^T \mathbf{M} \mathbf{S})^{-1}$ could be computed once and the result subsequently propagated back to the slave processors where the actual numerical integration is performed.

6 Performance Evaluation

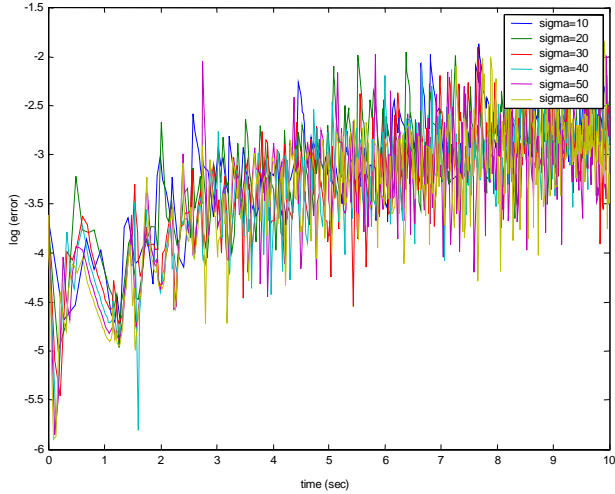
In the preceding section, we examined how both compliance-based and projection-based models of the four-bar were subdivided into two-part systems to be run in a distributed manner on separate processors, requiring only the exchange of state information at every time instant. This was implemented using RT-LAB, a commercial off-the-shelf system for distributing the computation and a series of tests were performed to evaluate the performance of both methods of simulation. The numerical parameters of the system were chosen as shown in Table 1. The two primary metrics of performance evaluation were: (a) extent of the constraint error; and (b) number of iterations/computational time required for each method. This performance was evaluated using two separate scenarios: adaptive time-stepping and fixed time-stepping.

Link lengths	$l_1 = 1.0\text{m}$	$l_2 = 2.5\text{m}$	$l_3 = 3.0\text{m}$
Distance of mass centers	$l_{c1} = l_1/2$	$l_{c2} = l_2/2$	$l_{c3} = l_3/2$
Link masses	$m_1 = 1.0\text{ Kg.}$	$m_2 = 1.0\text{ Kg.}$	$m_3 = 1.0\text{ Kg}$
Mass moment of Inertias	$I_1 = m_1 l_1^2/12$	$I_2 = m_2 l_2^2/12$	$I_3 = m_3 l_3^2/12$
Initial Conditions ($q(0)$)	$\theta = \pi/2$	$\alpha = 0.35328$	$\phi = 1.26486$
Torque inputs (u)	$\tau_1 = 6.0\text{Nm}$	$\tau_2 = 0.0\text{Nm}$	$\tau_3 = 0.0\text{Nm}$

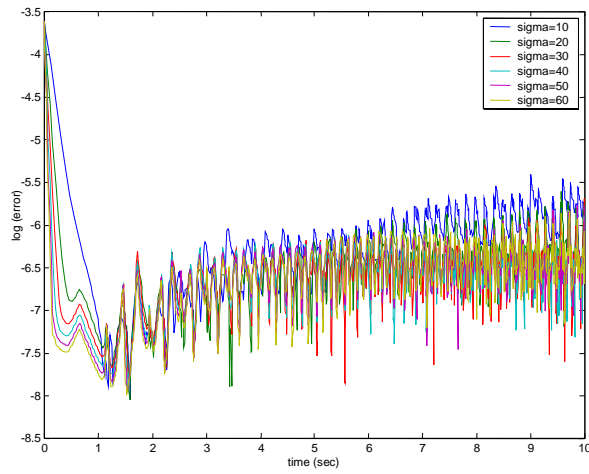
Table 1: Relevant numerical parameters for the four-bar linkage (Wang *et al.*, 2000)

6.1 Adaptive Time-stepping Scenario

In this scenario, the relative tolerance was pre-specified and an adaptive timestepping scheme is used for the simulation. Four different relative tolerances, varying in orders of magnitude from $1e-3$ to $1e-6$, were examined in this scenario. Each method has one independent parameter that could potentially affect the performance of the method – the stiffness of the virtual spring (K) in the compliance-based approach and the convergence factor (σ) in the case of the projection-based method. In the plots that result, we examine: the role of the independent parameter on the constraint error; and the effect of the independent parameter on the number of time-steps required to simulate a fixed simulation duration.



(a)

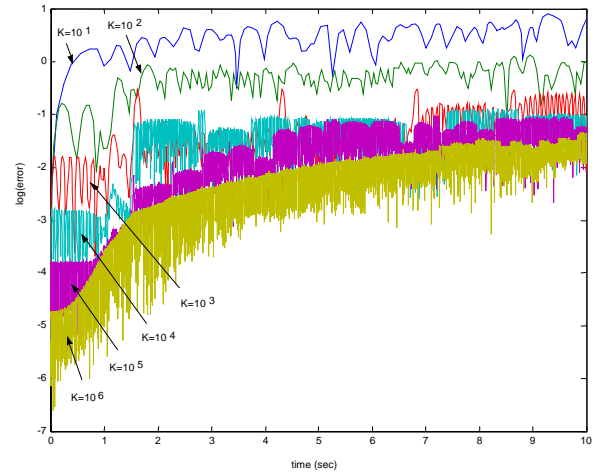


(b)

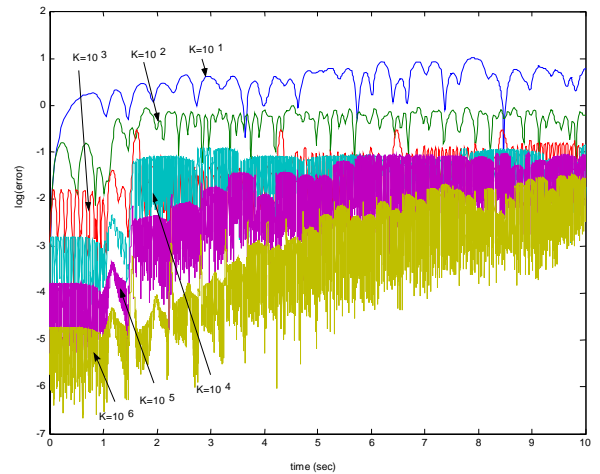
Figure 4: Constraint error in the projection-based approach using adaptive time-stepping for two sets of relative tolerances (a) $1e-3$ and (b) $1e-6$.

Figure 4 depicts the constraint error versus time for implementing the projection-based approach with different values of convergence factor (σ) for two sets of relative tolerances ($1e-3$ and $1e-6$) using the ODE45 Dormand-Prince adaptive time step solver. As can be seen from the results, the convergence factor (σ) does not significantly affect the constraint errors and that the resulting relative constraint errors are representative of the selected relative tolerance settings.

Figure 5 shows the resulting constraint errors that result from implementing the compliance-based approach with



(a)

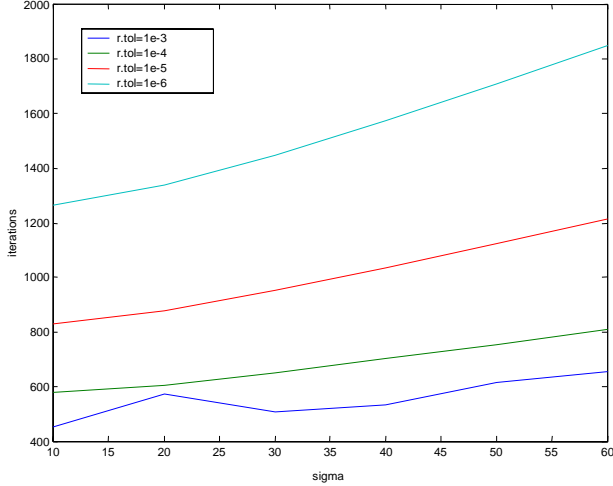


(b)

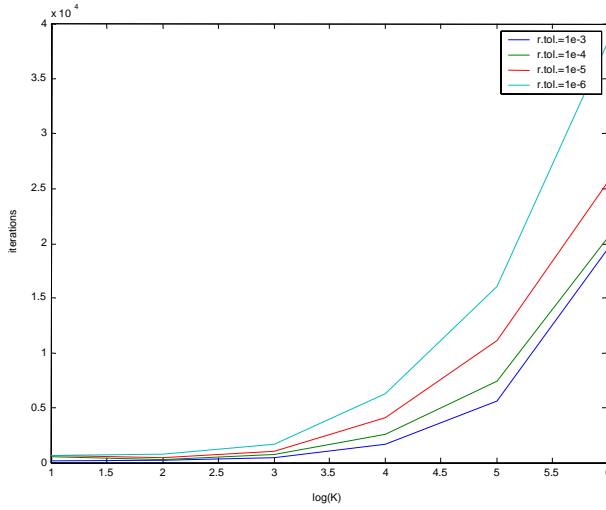
Figure 5: Constraint error in the compliance-based approach using adaptive timestepping for two sets of relative tolerances (a) $1e-3$ and (b) $1e-6$.

different values of the spring constant (K) (varied in orders of magnitude from $1e1$ to $1e6$).

These were carried out for various settings of the relative tolerances of the adaptive time step solver but only shown here for two cases ($1e-3$ and $1e-6$) in Fig. 5 (a) and (b) respectively. Qualitatively speaking, we note that decreasing the relative tolerance by three orders of magnitude did not affect the relative constraint errors, as can be seen by comparing Fig. 5 (a) and (b). However, within each graph, we note that the spring stiffness is increased, the constraint error drops commensurately but never attaining the performance levels of the projection-based method in Fig. 4.



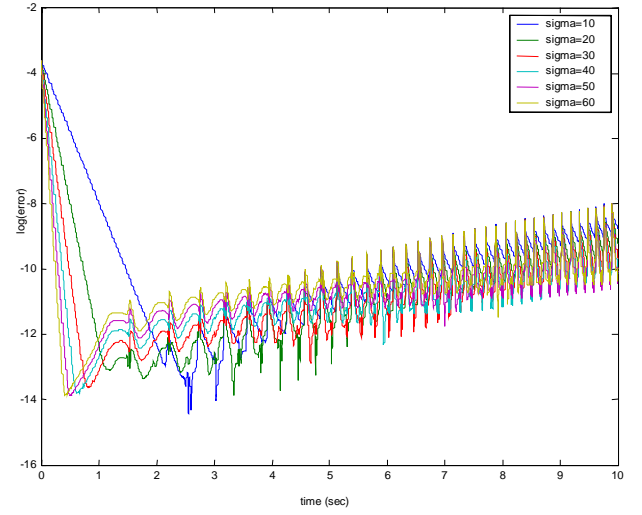
(a)



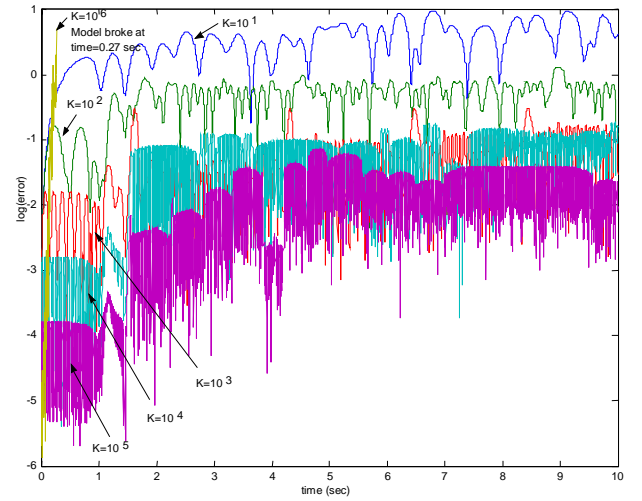
(b)

Figure 6: Number of iterations vs independent parameter for numerical integration of a 10 second duration with variable timestep for (a) Projection-based and (b) Compliance-based approaches.

To further compare the two methods in the adaptive time-stepping scenario, we examine the computational time required as measured in the number of discrete time-steps required to simulate 10 seconds. Figure 6 depicts the number of discrete time-steps of the adaptive time-stepping algorithm required to keep the constraint errors within the specified tolerances for different values of the parameters in each of the two methods. Two important features are to be noted: *First*, for increasing values of the parameter (σ or K) both methods show a distinct increases in the number of time-steps required to maintain a desired tolerance. However, while the rate of increase is linear in the case of the projection-based approach but tends to increase exponentially with increasing values of the spring stiffness in the compliance-based approach. *Secondly*, for a fixed value of



(a)



(b)

Figure 7: Constraint error for numerical integration with fixed timestep (1e-3 secs) for (a) Projection-based and (b) Compliance-based approaches.

the independent parameter, an exponential increase in the number of iterations can be seen with decreasing values of the relative tolerance in the compliance-based approach.

6.2 Fixed Time-stepping Case

This second scenario is more suited for real-time simulation and hardware-in-the-loop type simulation applications where deterministic time-stepping of the simulation is desirable. A number of simulations with different values for fixed time-steps (ranging from 1e-3 to 1e-6) were performed. However, only the resulting constraint errors from running the two simulation (compliance- and projection-based) schemes for a fixed step size of 0.001 seconds are shown in Fig. 7. In Fig. 7(a), we note that the selection of the value of the independent parameter (σ) only plays a minor role since regardless of the selected value the constraint error remains near about 1e-12. In contrast, in Fig. 7(b) we see that for

small values of the spring stiffness, considerable constraint error results which decreases as K is increased. While this constraint error reduces to the order of $1e-3$ as the spring stiffness is increased to $1e6$, this improvement is not comparable to the error magnitudes observed for the projection-based method.

7 Discussion

Schiehlen *et al.* (2000) performed a similar comparison by numerical simulation of a non-minimal description of a mechanical system obtained by coupling two or more minimal local subsystems with explicitly- or implicitly-stated holonomic constraints, with a variety of examples. They examined the effects of numerical integration with two similar classes of solution approaches, *i.e.* a so-called *force coupling approach*, which approximate the constraint forces by force laws proportional to the violation of coupling conditions; and the more traditional *differential algebraic approach*, using various integrators using projections on the position and velocity constraint manifold for stabilization.

As they note in their discussion, force coupled systems are especially sensitive to initial violations of the assembly conditions which may cause the coupling forces to create additional disturbances that excite the system. Further, such force-coupled systems can cause the dynamic behavior to be severely changed and even resulting in instability of the whole system. In contrast, they note that DAE description combined with integrators that use projection methods offer an efficient way of coupling local subsystems on the level of their equations of motion or for closing kinematic loops. Our own results, developed in the context of distributed computation of similar types of constrained dynamical systems match these observations.

In addition to the “natural” decoupling noted for the compliance-based formulation, several other advantages have been reported in the literature. These include the fact that appearance and disappearance of the constraints can be treated automatically and that the method performs robustly near kinematic singularity positions (Garcia de Jalón and Bayo, 1994). However, as noted by one of the reviewers, the Lagrange multipliers only form a part of the complete picture regarding the constraint forces. They represent the magnitude-type contribution while the other (and perhaps most important part) is the directional information that is embedded in the constraint Jacobian. The imperfect approximation of the Lagrange multipliers, coupled with the (artificial) relaxation of the constraints can over time lead to alternate configurations thereby indirectly affecting the directions of constraint vectors. Hence, not withstanding the small magnitudes of the constraint violations, the incorrect projection of the Lagrange multipliers would: yield seemingly correct but non-physical results; and additionally act as a continuous source of disturbance.

8 Summary

In this paper, we examined aspects of both the development and performance-evaluation of two alternate methods for distributed forward dynamics simulations of constrained mechanical systems exemplified by the four-bar linkage. Similar situations may also be encountered in other arenas where the governing equations take the form of sets of ODEs coupled together by algebraic constraints and solution of the combined system of DAEs needs to be found. We exploited the natural spatial parallelism of closed-chain manipulators initially for the modular development of overall dynamics and subsequently for the distributed numerical simulation of the dynamics. Traditionally, the numerical simulation problem has been treated as being composed of two more-or-less independent stages: an initial algorithm development stage followed by a numerical integration stage. While, the coupling of the two stages may not be as relevant for unconstrained mechanical systems, it plays a significant role for the simulation of constrained mechanical systems. Our preliminary results (examined in the context of four-bar linkage discussed in the previous section) indicates that a global unified view of the evaluation of the computational complexity of the simulation is advisable. Specifically, at an algorithmic development level, the compliance-based approach provides a seemingly natural method for decoupling and distributing the computation and has roughly one-third of the computational complexity of the projection-based approach. However, while the projection-based approach is computationally more expensive per time-step, fewer time-steps are required to maintain a desired relative constraint error tolerance leading to faster simulations.

Acknowledgements

We are indebted to the reviewers for their careful examination of the manuscript and their insightful comments that have helped to enhance the presented work. We would also like to acknowledge the support of the Natural Sciences and Engineering Research Council (NSERC) for support of this work.

9 References

- Arnold, V. 1989, *Mathematical Methods of Classical Mechanics*, 2nd ed. New York: Springer-Verlag.
- Ascher, U., Chin, H., Petzold, L., and Reich, S., 1995, “Stabilization of constrained mechanical systems with DAEs and invariant manifolds,” *Mechanical Structures & Machines*, 23:135-158.
- Ascher, U., Pai, D. K., and Cloutier, B., 1997, “Forward dynamics, elimination methods, and formulation stiffness in robot simulation,” *The International Journal of Robotics Research*, 16(6):749-758.

- Ascher, U., and Petzold, L., 1998, *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*, SIAM.
- Ascher, U. and Lin, P., 1999, "Sequential regularization methods for simulating mechanical systems with many closed loops," *SIAM Journal of Scientific Computing*, 21:1244-1262.
- Baumgarte, J., 1983, "A new method of stabilization for holonomic constraints," *ASME Journal of Applied Mechanics*, 50:869-870.
- Blajer, W., Bestle, D., Schiehlen W., 1994, "An orthogonal complement matrix formulation for constrained multibody systems," *ASME Journal of Mechanical Design*, Vol.116, pp. 423-428.
- Chaudhary, V. and Aggarwal J. K., 1990, "Parallelism in computer vision: A review", in *Parallel Algorithms for Machine Intelligence and Vision*, Springer-Verlag:New York.
- Featherstone, R., 1983, "The calculation of robot dynamics using articulated-body inertias," *The International Journal of Robotics Research*, 2(1):13-30.
- Featherstone, R., 1987, *Robot Dynamics Algorithms*, Kluwer.
- Fijany, A. and Bejczy, A. K., 1992, *Parallel Computation Systems for Robotics: Algorithms and Architectures*, World Scientific, 1992.
- García de Jalón, J., and Bayo, E., 1994, *Kinematic and Dynamic Simulation of Multibody Systems: The Real-Time Challenge*, Springer-Verlag, New-York.
- Haug, E.J., 1989, *Computer-Aided Kinematics and Dynamics of Mechanical Systems*, Allyn and Bacon.
- Henrich, D., and Höniger T., 1997, "Parallel processing approaches in robotics," Proceedings of the *IEEE International Symposium on Industrial Electronics (ISIE'97)*, Guimaraes, Portugal, pp 702-707.
- Kecskemethy, A., Krupp, T., and Hiller, M., 1997, "Symbolic processing of multi-loop mechanism dynamics using closed form kinematic solutions," *Multibody System Dynamics*, 1(1):23-45.
- Lee, C. S. G. and Chang, P. R., 1986, "Efficient parallel algorithms for robot inverse dynamics computation," *IEEE Transactions on Systems, Man, and Cybernetics*, 16:532-542.
- Lee, C. S. G. and Chang, P. R., 1988, "Efficient parallel algorithms for robot forward dynamics computation," *IEEE Transactions on Systems, Man, and Cybernetics*, 18:238-251.
- McMillan, S., 1994, *Computational Dynamics for Robotic Systems on Land and Under Water*, Ph.D. Thesis, The Ohio State University, Columbus, OH.
- Murray, R., Li, Z., and Sastry, S. S., 1994, *A Mathematical Introduction to Robotic Manipulation*, CRC Press.
- Sadayappan, P., Ling, Y. L. C., Olson, K. W., and Orin, D. E., 1989, "A restructurable VLSI robotics vector processor architecture for real-time control," *IEEE Transactions on Robotics and Automation*, 5:583-599.
- Serna, M.A., Avilés, R., and García de Jalón, J., 1982, "Dynamic Analysis of Plane Mechanisms with Lower-Pairs in Basic Coordinates", *Mechanism and Machine Theory*, Vol. 17, pp. 397-403.
- Schiehlen, W., 1990, *Multibody Systems Handbook*, Springer-Verlag, Berlin.
- Schiehlen W., Rukgauer A., and Schirle, Th., 2000, "Force Coupling versus Differential Algebraic Description of Constrained Multibody Systems", *Multibody System Dynamics*, 4(4):317-340.
- Shabana, A.A., 1989, *Dynamics of Multibody Systems*, Wiley, New York.
- Walker, M. W. and Orin, D. E., 1982, "Efficient dynamic computer simulation of robotic mechanisms," *ASME Journal of Dynamic Systems, Measurement, and Control*, 104:205-211.
- Wang, J., Gosselin, C.M. and Cheng, L., 2000, "Dynamic modeling and simulation of parallel mechanisms using the virtual spring approach", Proceedings of *2000 ASME Design Engineering Technical Conferences*, Baltimore, Maryland, DETC2000/MECH-14107.
- Yun, X. and Sarkar, N., 1998, "A unified formulation of robotic systems with holonomic and non holonomic constraints," *IEEE Transactions on Robotics and Automation*, 14 (4): 640-650.
- Zoyama A. B., 1993, *Modelling and Simulation of Robot Manipulators: a parallel processing approach*, Vol. 8, World Scientific Series in Robotics and Intelligent Systems. World Scientific.