Distributed Realization of Differential-Algebraic Systems Using Decentralized Sliding Mode Control

Seyyedmohsen Azizi

A Thesis
in
The Department
of
Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of Master of Applied Science (Electrical Engineering) at
Concordia University
Montreal, Quebec, Canada

December, 2005

Your file Votre référence
ISBN: 0-494-14253-7
Our file Notre référence
ISBN: 0-494-14253-7

Abstract

Seyyedmohsen Azizi

Distributed Realization of Differential-Algebraic Systems Using Decentralized Sliding Mode Control

Differential-algebraic Equation (DAE) systems present numerous difficulties in distributed simulation and control systems. The main problem is that most existing methods require an explicit state space model without algebraic constraints. One approach to address this problem is to reformulate the DAE system into an equivalent nonlinear control problem, in which the algebraic constraints are replaced by appropriate sliding manifolds. However, previous approaches based on this method are centralized leading to a great deal of computation and communication in distributed environments associated with inversion of the associated input decoupling matrix. In this work, this problem is addressed through application of decentralized sliding mode control. Relationships are developed for stability and performance in the presence of the neglected coupling terms. Inversion of the decoupling matrix is performed on a node basis. This allows the systematic division of the system into nodes that are more efficient for distributed computation. The new approach is applied to simulation of deformable surfaces in a real-time distributed computing environment.

Acknowledgments

This thesis was carried out in the Control and Information System Laboratory (CIS) at Concordia University, Montreal, Canada. It is my honor to be a part of this dynamic group of researchers.

I would like to thank my supervisors, Dr. Brandon Gordon and Dr. Venkat Ramachandran, for all their supports throughout my academic life so far.

I would like to express my respects and regards to my wife and parents, as their everlasting love and encouragement are the warm and shining sources of my life. Their motivation and support were the spirit of all the moments in my work.

"Education is what remains after one has forgotten everything he learned in school."

- Albert Einstein

Table of Contents

List of Figures

Nomenclature

r	Index
$\mathbf{r}$	Vector index
$\mathbf{w}$	Output vector
$\mathbf{v}$	Input vector
s_i	i-th sliding surface
$-\frac{1}{\mu}$, $\mu > 0$	Characteristic equation poles of the sliding surface
n	Dimension of the state vector
m	Dimension of the input and output vectors
ε	Boundary layer thickness
$\boldsymbol{\delta}$	Relative importance vector of sliding surfaces
$\mathbf{K}$	Gain Matrix
$\mathbf{J}_s$	Jacobian Matrix
$\hat{\mathbf{J}}_s$	Computationally efficient approximation of Jacobian Matrix
$\mathbf{W}_b$	Set of boundary algebraic equations
$\mathbf{v}_b$	Boundary input vector of the boundary node
$\mathbf{w}_b$	Boundary output vector of the boundary node
L	Number of boundary algebraic equations

$\mathbf{v}_i$	Local input vector of the i-th local node
$\mathbf{w}_i$	Local output vector of the i-th local node
N	Number of local nodes
$\widetilde{\mathbf{J}}_{sb}$	Jacobian matrix of the boundary node
$\widetilde{\mathbf{J}}_{si}$	Jacobian matrix of the i-th local node
$\boldsymbol{\theta}_i$	Interconnection vector of the i-th local node
$\mathbf{d}_b$	Uncertainty and disturbance vector of the boundary node
$\boldsymbol{\eta}$	Internal dynamic states
n_i	Dimension of the state vector of the i-th subsystem
m_i	Dimension of the input and output vectors of the i-th subsystem
N_i	Dimension of the state vector of the i-th local node
M_i	Dimension of the input and output vectors of the i-th local node
N_b	Dimension of the state vector of the boundary node
M_b	Dimension of the input and output vectors of the boundary node
n	Number of the subsystems
r_i	The equal elements of the vector index of the i-th local node
r_b	The equal elements of the vector index of the boundary node

$\mathbf{E}_{\alpha}$	Estimation error of $\boldsymbol{\alpha}$
$\mathbf{E}_{\Delta\Theta}$	Estimation error of $\Delta\boldsymbol{\Theta}$
$\mathbf{E}_{\Delta D}$	Estimation error of $\Delta\mathbf{D}$
$\mathbf{B}_{\Delta\theta}$	Bound of $\Delta\boldsymbol{\Theta}$
$\mathbf{B}_{\Delta D}$	Bound of $\Delta\mathbf{D}$
$\boldsymbol{\Psi}_{b}$	Adaptive coefficient matrix
$\boldsymbol{\psi}_{bj}$	Adaptive coefficient vector
$\mathbf{F}(t)$	Basis function vector
ε_{i}	Boundary layer thickness of the i-th local node
ε_{b}	Boundary layer thickness of the boundary node
$\boldsymbol{\Omega}_{1}$	Left diagonal adaptation rate matrix
$\boldsymbol{\Omega}_{2}$	Right diagonal adaptation rate matrix
ω_{ij}	Adaptation rate elements
m_{i}	i-th particle mass
$\vec{\mathbf{F}}_{int,i}$	Internal force to the i-th particle mass
$\vec{\mathbf{F}}_{ext,i}$	External force to the i-th particle mass
$\hat{r}_{ij}$	Unity vector from i-th particle to j-th particle
L_{k}	Length of the k-th link
L_{k0}	Desired length of the k-th link

$-\frac{1}{\mu_i}$, $\mu_i > 0$	Characteristic equation poles of the sliding surface of the i-th local node
$-\frac{1}{\mu_b}$, $\mu_b > 0$	Characteristic equation poles of the sliding surface of the boundary node
$\boldsymbol{\delta}_i$	Relative importance vector of the i-th local node
$\boldsymbol{\delta}_b$	Relative importance vector of the boundary node
$\boldsymbol{\kappa}_i$	Gain vector of the i-th local node
$\boldsymbol{\kappa}_b$	Gain vector of the boundary node
γ	Stability margin
γ_i	Stability margin of the i-th local node
γ_b	Stability margin of the boundary node

1 . Introduction

1.1 Motivation

Differential Algebraic Equation (DAE) systems provide a more general description of dynamical systems than ordinary differential equations (ODEs). However DAE systems present a number of difficulties in simulation and control. The main problem is that most methods require an explicit state variable model. One method to address this problem is to reformulate the DAE system into an equivalent nonlinear control problem, in which the algebraic constraints are satisfied by sliding manifolds. This approach results in a state space approximation to the DAE. A robust sliding mode controller can be designed to achieve a reasonable approximation error. However, when the order of the system is very large, it is computationally too expensive to control the system on a single computer. The sliding mode simulation method is inherently centralized due to the input decoupling Jacobian matrix which must be inverted at each time step. This can cause significant problems since the matrix inversion problem is difficult to divide onto multiple processors.

In this work, the problem mentioned above is investigated, and a method based on decentralized sliding mode is proposed to apply to the distributed state space realization of the DAE. After defining the set of boundary algebraic equations, the system is decoupled into many nodes by breaking along these connective algebraic equations. First, a separate sliding mode controller is designed for each node on a separate computer. Next, an extra integrator is applied to each control input of the connective boundary algebraic equations, and separate sliding mode controllers are designed for them on a separate computer. Because the internal dynamic of the system is stable, the decentralized sliding controller makes the whole system stable. This method makes the decentralized control possible, while eliminating the interconnection between the distributed computers, which in turn substantially decreases the information flow rate. This approach is applicable to a variety of problems including power network simulation, virtual reality, and hardware in the loop simulation.

1.2 Literature Review

Generally, the dynamic systems could be classified into two categories: ordinary differential equation (ODE) and differential-algebraic equation (DAE) systems. As it could be understood from the name, a DAE system is a set of differential equations constrained by a set of algebraic equations. The algebraic equations are the result of modeling simplifications such as steady state approximation of fast dynamics and rigid body assumptions that result in kinematic constraints. Hence, DAE systems represent a more general class of dynamic systems and are capable of behaviors that are fundamentally different from conventional ODE systems.

Simulation of dynamic systems is of interest in different aspects. There are not too many difficulties involved with simulation of ODE systems. But for DAE systems, the situation is quite different. Since a DAE system could simply be a piece of cloth, chain, hair or jelly-type materials, there are some challenging issues involved with the nature of these types of materials. In all these problems, we are dealing with a kind of stiffness in some directions. Despite the generality of DAE systems and their simplification of the modeling process, few methods currently exist for simulation and control of this class of systems. Since the constraints are mostly nonlinear, they cannot be eliminated easily. Therefore, the main focus is to look for methods that address the DAE systems directly.

The simulation of DAE systems could be extended to a variety of problems including power network simulation, virtual reality, and hardware in the loop simulation. In all these problems, the set of algebraic equations impose some constraints on the set of differential equations. These constraints are the algebraic models describing the conservation of parameters such as length and energy.

The traditional methods [39] cannot handle the simulation of a DAE system efficiently, because the set of equations are stiff and the time step should be chosen small enough. The alternative approach is to solve the equations by imposing and correcting the constraints [40]. This method requires further calculations involved with the algorithms to correct the momentum and energy alterations, which is a direct result of constraint enforcement [41]. Hence, it is more costly in terms of calculation time.

The proposed approach in [4] has a built-in mechanism to handle the limits on the constraints, and solve the set of ODEs at the same time. In this method, the DAE system is considered as a state space system, where the differential equations are the ones corresponding to the state update equations, and the algebraic equations are simply taken as the output. The objective of the control problem is to control the output to converge to zero, so that the algebraic constraints are satisfied. The control method implemented to design the controller is the sliding mode control. This method is quite efficient in terms of stability and approximation error. Moreover, this method is accompanied by the singular perturbation theorems. This combination helps to determine the realization error convergence. It also helps to select the parameters of the sliding controller in a way that the error is bounded in a desired range.

In the sliding control and singular perturbation approach, there is a matrix inversion problem corresponding to the Jacobian matrix. The case for a full matrix with no sparseness can be used as an upper bound for the matrix inversion time. The dimensions of this Jacobian matrix increases according to the number of the algebraic constraints. Therefore, calculating the inverse of the Jacobian matrix becomes so time consuming that it violates the timing limitations of a real time simulation.

So far, the focus was on the parallel processing methods for a distributed simulation problem [42,43,44,45,46]. These methods are all based on using some computers and processors in parallel, and implementing the specific computer programs and algorithms to share the time consuming computations, like the inversion of the large-dimension

Jacobian matrix, between all the processing units in parallel [47,48,49]. These methods are considered as centralized, since all the processing units are directly in connection with each other. The whole simulation system is not robust to any failure that may happen in the function of one unit. Therefore, all control parameters should be chosen and designed with the highest stability margin. This makes the controller designing procedure difficult, and the designed controller leads to a system which is not robust enough.

In order to get rid of the problems involved with the centralized controller, the best remedy is to replace the centralized control method by the decentralized one [15]. In this work, we are concerned with the sliding control method for the decentralized case. The objective is to replace the centralized control method with the decentralized one. Mainly, this method pertains to the applications in which a large scale system is concerned. Meanwhile, in the distributed simulation problem, the main timing problem emerges when the dimensions of the dynamic system become too large. As a result, distributed sliding controller is the best match to address the timing problems.

A group of decentralized sliding control methods are based on Fuzzy methods [6,9,12,13,19]. In these methods some fuzzy estimation are used in order to cancel out some unknown terms related to the uncertainties, disturbances and unmodelled dynamics, so that the sliding input will guarantee the Lyapunov stability [23,24,26,27,29,34].

In [7,14,35], the decentralized sliding control methods are based on adaptive estimation of the unknown terms including the uncertainties, modeling errors and disturbances. In these methods, it is assumed that the unknown terms are approximated or bounded by a finite summation of power terms. On the other hand, in [36] these terms are approximated by the Fourier series expansion with finite number of terms, and adaptive laws are applied to adjust the Fourier coefficients.

Distributed realization of DAE systems, which is presented in this work, is a new modeling approach that solves the timing problem with Jacobian matrix inversion. It is a distributed modeling approach that allows the DAE model to be distributed on a networked computational cluster. The distributed realization method is based on state space realization of DAE systems and variable structure control. Therefore, the following subsections explain about these issues in detail.

1.2.1 Differential-Algebraic Equation (DAE) Systems

In this section, nonlinear DAE systems are introduced, and the control method to simulate the system, while satisfying the algebraic constraints, is formulated.

A Differential Algebraic Equation (DAE) is a nonlinear system expressed as:

$$\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}, \mathbf{z}) \qquad (1)$$

$$\mathbf{0} = \mathbf{g}(t, \mathbf{x}, \mathbf{z}) \qquad (2)$$

where $\mathbf{x} \in R^n$, $\mathbf{z} \in R^m$, $\mathbf{f}: R \times R^n \times R^m \to R^n$, and $\mathbf{g}: R \times R^n \times R^m \to R^m$. Equations (1) and (2) are sufficiently differentiable, and a well-defined solution with consistent initial conditions exists for x and z. The index of a DAE, which is a structural property of the system, is defined below [11].

Definition 1. The minimum number of times that all or part of the algebraic equations (2) must be differentiated with respect to time in order to solve for $\dot{\mathbf{z}}$ as a continuous function of t, x, and z is the index of the DAE system.

Differentiating equation (2) once results in:

$$\mathbf{0} = \frac{\partial \mathbf{g}}{\partial t} + \frac{\partial \mathbf{g}}{\partial \mathbf{x}}\mathbf{f} + \frac{\partial \mathbf{g}}{\partial \mathbf{z}}\dot{\mathbf{z}} \tag{3}$$

If the Jacobian $[\partial \mathbf{g} / \partial \mathbf{z}]$ is nonsingular, then the index of the system is one, and equation (3) could be solved for $\dot{\mathbf{z}}$. If the system index is of higher order (index>1), then the Jacobian matrix is not invertible and the constraint equations are identically singular with respect to **z**. Therefore, a more complex method is applied for high index DAE systems.

1.2.2 State Space Realization of DAE Systems

For the realization purposes, a high index DAE system can be reformulated as an equivalent nonlinear system, by ignoring the algebraic equations, while considering them as the outputs to be controlled, in order to converge to zero within a reasonable settling time. In this case, the nonlinear control theory could be applied to solve the realization problem.

For the DAE in equations (1) and (2) above, the equivalent nonlinear system is:

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{f}(t,\mathbf{x},\mathbf{z})\\ \dot{\mathbf{z}} &= \mathbf{v}\\ \mathbf{w} &= \mathbf{g}(t,\mathbf{x},\mathbf{z})\end{aligned} \tag{4}$$

where $\mathbf{v}$ is the virtual input that drives $\mathbf{z}$, which is assumed to be an independent parameter, and $\mathbf{w}$ is an output equal to the violation of the constraint algebraic equations. The relative degree of this problem is equal to the index of the equivalent DAE, and the zero dynamics represent the dynamics of the high index DAE. As a result, the state space realization of a high index DAE system can be interpreted as the combination of the system (4) and a controller which forces the outputs to zero (figure 1).

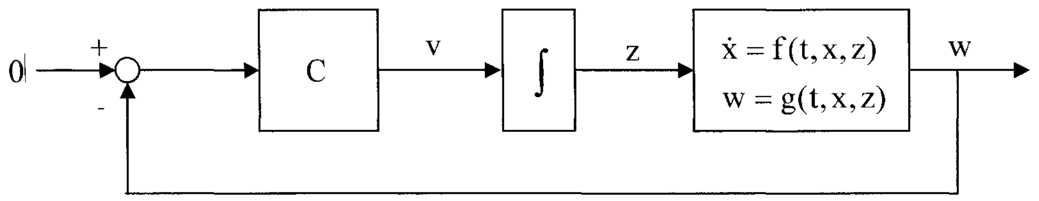


Figure 1. State space realization of a DAE.

In general, each component w_i of the constraint equations may need to be differentiated a different number of times ($r_i - 1$) for some components of z to appear. That is,

$$\begin{aligned}0 &= w_i(t,\mathbf{x})\\ 0 &= \frac{dw_i}{dt}(t,\mathbf{x})\\ &\vdots\\ 0 &= \frac{d^{r_i-1}w_i}{dt^{r_i-1}}(t,\mathbf{x},\mathbf{z})\ ,\ 1\le i\le m\end{aligned} \tag{5}$$

For z to be explicitly determined by the constraints (5), the following Jacobian matrix should be nonsingular.

$$\mathbf{J}_{\Omega} = \frac{\partial \mathbf{\Omega}}{\partial \mathbf{z}} \tag{6}$$

where

$$\Omega_i = \frac{d^{r_i - 1} w_i}{dt^{r_i - 1}}(t, \mathbf{x}, \mathbf{z}) \;,\; 1 \le i \le m \tag{7}$$

According to equation (7), the constraints should be differentiated r_i times to determine $\dot{\mathbf{z}}$ which is equal to the virtual input $\mathbf{v}$.

$$\mathbf{0} = \frac{\partial \mathbf{\Omega}}{\partial t} + \frac{\partial \mathbf{\Omega}}{\partial \mathbf{x}}\mathbf{f} + \frac{\partial \mathbf{\Omega}}{\partial \mathbf{z}}\dot{\mathbf{z}} \tag{8}$$

Definition 2. If the Jacobian $\mathbf{J}_{\Omega}$ is nonsingular in a region around the DAE, then the vector index is defined by

$$\mathbf{r} \equiv [r_1 \quad \ldots \quad r_m]^T \tag{9}$$

where r_i is the number of times each constraint w_i must be differentiated for components of $\dot{\mathbf{z}}$ to appear. The differentiations in (7) will explicitly determine $\dot{\mathbf{z}}$. For many types of physical systems, it can be shown that the Jacobian is always nonsingular, and the index is defined as the largest component of the vector index (9). In the next subsection an effective nonlinear control approach is developed to apply to the state space and make an overall equivalent realization of the primary DAE system.

1.2.3 Variable Structure Control

In this subsection, a nonlinear controller is designed to force the state space system (4) approximately converge to the DAE system {(1),(2)} in the presence of system uncertainties. The main objectives are to force the outputs in (4) to zero in order to meet the constraints in (2), to reduce the computational complexities, and to achieve reasonable approximation errors. For designing the sliding manifolds, the following theorem is considered:

Theorem 1. For a DAE system with the vector index r, the sliding manifold s can be constructed with the elements defined as:

$$s_i = \left(\mu \frac{d}{dt} + 1\right)^{r_i - 1} w_i \ , \ \mu > 0 \ , \ 1 \le i \le m \tag{10}$$

where s needs to be differentiated only once to determine $\dot{\mathbf{z}}$ $(= \mathbf{v})$, and $\partial \mathbf{s} / \partial \mathbf{z}$ is non-singular around the DAE solution. The equation $\mathbf{s} = \mathbf{0}$ has an attractive invariant set composed of the algebraic equations (2).

Proof. See reference [4].

□

The equation $\mathbf{s} = \mathbf{0}$ consists of m differential equations, each corresponding to a stable polynomial in the Laplace domain. Therefore, the dynamic system (4) is exponentially stable, and so the algebraic equations (2) are asymptotically satisfied. The equation $\mathbf{s} = \mathbf{0}$ has an invariant set composed of the high index constraints. This implies that once the algebraic equations (2) are satisfied with some reasonable errors, they will remain within the same error bounds, and so a solution to the high index DAE is achieved with a reasonable accuracy.

After designing the sliding manifolds in the section above, the objective is to force them to zero by a sliding mode controller. The following quadratic Lyapunov function is considered:

$$V = \frac{1}{2}\mathbf{s}^T\mathbf{s} > 0 \tag{11}$$

The derivative of this function is

$$\frac{dV}{dt} = \mathbf{s}^T\dot{\mathbf{s}} = \mathbf{s}^T\left(\boldsymbol{\alpha} + \mathbf{J}_s\mathbf{v}\right) \tag{12}$$

where

$$\mathbf{J}_s = \frac{\partial \mathbf{s}}{\partial \mathbf{z}} \quad , \quad \boldsymbol{\alpha} = \frac{\partial \mathbf{s}}{\partial t} + \frac{\partial \mathbf{s}}{\partial \mathbf{x}}\mathbf{f} \tag{13}$$

Now, the following control law is applied:

$$\mathbf{J}_s \cdot \mathbf{v} = -\hat{\boldsymbol{\alpha}} - \mathrm{diag}(\boldsymbol{\kappa})\mathbf{sgn}(\mathbf{s}) \tag{14}$$

where $\hat{\boldsymbol{\alpha}}$ is an efficiently computational approximation of the term $\boldsymbol{\alpha}$, $\boldsymbol{\kappa}$ is a gain vector, and **sgn** is a component-wise sign function. According to theorem 1, the Jacobian $\mathbf{J}_s$ is locally nonsingular, and so it can be solved for the required input, explicitly. Replacing equation (14) in (12) results in:

$$\frac{dV}{dt} = \mathbf{s}^T(\boldsymbol{\alpha} - \hat{\boldsymbol{\alpha}}) - \sum_{i=1}^{m} \kappa_i |s_i| \tag{15}$$

The elements of the vector $\boldsymbol{\kappa}$ should be chosen as:

$$\kappa_i \geq |\alpha_i - \hat{\alpha}_i| + \gamma_i \ , \ \gamma_i > 0 \ , \ 1 \leq i \leq m \tag{16}$$

so that the following sliding condition for stability is satisfied:

$$\frac{dV}{dt} = \frac{1}{2}\frac{d}{dt}\mathbf{s}^T\mathbf{s} \leq -\sum_{i=1}^{m} \gamma_i |s_i| \tag{17}$$

and the algebraic equations (2) will be satisfied. That is because $\mathbf{s} = \mathbf{0}$ is an invariant set of the system and the algebraic equations form an invariant set within $\mathbf{s} = \mathbf{0}$. Therefore, once $\mathbf{s} = \mathbf{0}$ is held and the algebraic equations are satisfied within some reasonable error bounds, then they will always be satisfied. As a result, the sliding control method helps to make an exact representation of the DAE. The only problem with the control input (14) is the discontinuity originated from the **sgn** function. This is undesirable because a very small time step is required to discretize the system for simulation and implementation purposes. Moreover, this discontinuity helps the chattering behavior of the control input, which should be avoided specifically in real-time control. To overcome the discontinuity problem and smooth the input in a boundary layer region around the sliding surface, the sign function is replaced by a saturation function (figure 2) [5].

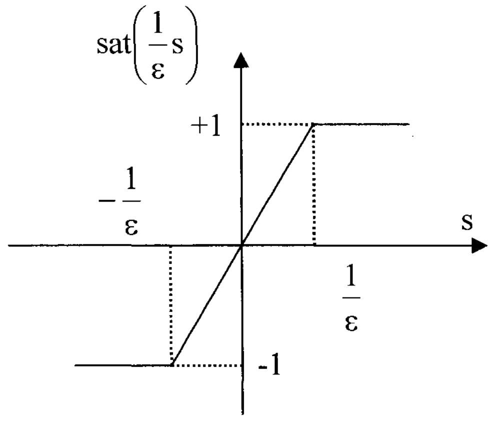


Figure 2. A saturation function.

Hence, the equations become into the form:

$$\begin{aligned}
&\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}, \mathbf{z}) \\
&\dot{\mathbf{z}} = \mathbf{v} \\
&\mathbf{w} = \mathbf{g}(t, \mathbf{x}, \mathbf{z}) \\
&\hat{\mathbf{J}}_s \mathbf{v} = -\hat{\boldsymbol{\alpha}} - \mathbf{K}\,\mathbf{sat}(\varepsilon^{-1}\mathrm{diag}(\boldsymbol{\delta})\,\mathbf{s}) \quad , \quad 0 < \varepsilon << 1 \\
&s_i = \left(\mu \frac{d}{dt} + 1\right)^{r_i - 1} w_i \;,\; \mu > 0 \;,\; 1 \le i \le m
\end{aligned} \tag{18}$$

where $\hat{\mathbf{J}}_s$ is a computationally efficient approximation of $\mathbf{J}_s$, $\mathbf{sat}$ is a component-wise saturation function, and $\boldsymbol{\delta}$ is a vector specifying the relative importance of each sliding surface. In general, the matrix $\mathbf{K}$ is not necessarily diagonal. In order to achieve an approximation with enough accuracy, this boundary layer should be thin enough accordingly. As a result, the dynamics become singularly perturbed, and the sliding control method is combined with the singular perturbation approach [2]. The condition for the realization (18) to be a good approximation of the DAE system is stated in theorem 2.

Theorem 2. If the matrix $[\mathbf{J}_s\hat{\mathbf{J}}_s^{-1}\mathbf{K} - \boldsymbol{\Gamma}]$ is diagonally dominant, then the sliding condition for stability is satisfied, which guarantees the following error bounds

$$|s_i| \le \frac{\varepsilon}{\delta_i} \ , \ 0 \le i \le m \tag{19}$$

and

$$\left|\frac{d^j w_i}{dt^j}\right| \le \frac{2^j \varepsilon}{\delta_i \mu^j} \ , \ 0 \le j \le r_i\text{-}1 \ , \ 0 \le i \le m \tag{20}$$

where $\boldsymbol{\Gamma} = \left| \operatorname{diag}[\boldsymbol{\alpha} - \mathbf{J}_s\hat{\mathbf{J}}_s^{-1}\hat{\boldsymbol{\alpha}}] \right|$

Proof. See reference [4].

□

According to this theorem, there is a choice of ε to provide error bounds with enough accuracy. After the time constant μ in the sliding manifolds is chosen, the parameter ε can be adjusted to meet the desired constraint tolerances. There is a trade-off between the control bandwidth and Jacobian computation. If more Jacobian terms are included in $\hat{\boldsymbol{\alpha}}$, then $\boldsymbol{\Gamma}$ is reduced and less control gain K is needed to satisfy the criteria in theorem 2. Furthermore, the error bounds in theorem 2 are theoretically important, because they are used to guarantee the stability of the realization within a specific region around the DAE solution, where the vector index is well defined.

1.3 The Contribution and Outline of the Thesis

The approach presented so far is a quite efficient method to simulate the mechanical DAE systems. The most significant problem is when m, the dimension of the algebraic equations (2), increases considerably. In this case, the dimension of the Jacobian matrix $\mathbf{J}_s$ increases accordingly. So computation of the inverse matrix $\hat{\mathbf{J}}_s^{-1}$ becomes too expensive for a single computer, specifically in the cases in which real time simulation is required. This expense pertains to the timing restriction of the simulation. The timing problem involved with the calculation of the inverse matrix is not the same for all the Jacobian matrices in all the simulation steps. The concern is mainly with the worst case that may occur occasionally during a simulation. The worst case for the matrix inversion problem occurs whenever the Jacobian becomes a full matrix with no sparseness. Hence, the calculation involved with this kind of Jacobian can be used as a bound for the matrix inversion time.

The method proposed in this work is a modeling approach for DAE systems with large scale dimensions. This method satisfies the timing requirements of a real-time simulation. The contribution of this thesis is as follows:

- Remodeling the DAE system and transforming it into the distributed formulation.

- Applying the decentralized sliding controller method.

Therefore, the main topics in the thesis can be outlined as follows:

- Distributed modeling of the DAE
- Applying decentralized sliding controller
- Parameter designing for the controller
- Application to animation
- Comparison with centralized simulation

The proposed distributed modeling approach puts a solution in front, to use a number of computers instead of a single computer, so that the timing requirements of a real-time simulation could be met. However, optimization of the simulation and partitioning is not in the scope of the thesis. This part of the problem is an important open area for further consideration. This issue could be investigated by engineers according to the type of application.

2 . Distributed Realization and Decentralized Control

In this section, a new approach is proposed to decouple the DAE system into multiple nodes. Each node can be simulated on a separate computer, while a decentralized controller is designed on each separate computer to control the corresponding local node. The method proposed in this chapter is for a fairly general class of DAE systems. However, it is assumed that the engineer has suitably partitioned the equations into a number of nodes and identified the boundary constraints between these nodes. This can be readily accomplished by an engineer with basic knowledge of the model structure.

2.1 Distributed Formulation of the DAE System

In order to reformulate a DAE system into the distributed mode, some expressions should be defined as the key tool to decouple the system. The key definition is the set of boundary algebraic equations.

2.1.1 Set of Boundary Algebraic Equations

Assumption 1. The main concern of this work is with the systems that have vector index with equal elements $r_1 = r_2 = \cdots = r_m = r$.

The DAE realization of a nonlinear system consisting of n subsystems is considered. The i-th subsystem (i=1,...,n) is described by the following equations:

$$
\begin{aligned}
&\dot{\mathbf{x}}_i = \mathbf{f}_i(\mathbf{x}_1,\mathbf{x}_2,\dots,\mathbf{x}_n) + \sum_{j=1}^{n} \mathbf{h}_{ij}(\mathbf{x}_1,\mathbf{x}_2,\dots,\mathbf{x}_n,\mathbf{z}_j) \\
&\dot{\mathbf{z}}_i = \mathbf{v}_i \qquad\qquad (i = 1,\dots,n) \\
&\mathbf{w}_i = \mathbf{g}_i(\mathbf{x}_1,\mathbf{x}_2,\dots,\mathbf{x}_n)
\end{aligned}
\tag{21}
$$

where $x_i \in R^{n_i}$, $\mathbf{v}_i \in R^{m_i}$ and $\mathbf{w}_i \in R^{m_i}$ are the state, input, and output vectors of the i-th subsystem, respectively. $\mathbf{f}_i$'s, $\mathbf{h}_{ij}$'s and $\mathbf{g}_i$'s are the vector functions with appropriate dimensions. Moreover, the dimensions of all the vectors and matrices are well defined. Considering assumption 1, suppose that this system has a vector index with equal elements $r_i = 3$, i.e.

$$
\frac{\partial \mathbf{g}_i}{\partial \mathbf{x}_k} \cdot \mathbf{h}_{kj} = 0 \tag{22}
$$

, the matrix $\mathbf{J}_s$ in the most general case is:

$$
\mathbf{J}_s = \begin{bmatrix} \mathbf{J}_{s11} & \mathbf{J}_{s12} & \cdots & \mathbf{J}_{s1n} \\ \mathbf{J}_{s21} & \mathbf{J}_{s22} & \cdots & \mathbf{J}_{s2n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{J}_{sn1} & \mathbf{J}_{sn2} & \cdots & \mathbf{J}_{snn} \end{bmatrix} \quad , \quad \det(\mathbf{J}_s) \neq 0 \tag{23}
$$

where

$$\mathbf{J}_{\mathrm{sij}} = \mu^2 \cdot \sum_{m=1}^{n} \left(\frac{\partial}{\partial \mathbf{x}_m} \sum_{k=1}^{n} \left(\frac{\partial \mathbf{g}_i}{\partial \mathbf{x}_k} \cdot \mathbf{f}_k \right) \frac{\partial \mathbf{h}_{mj}}{\partial \mathbf{z}_j} \right) \tag{24}$$

When the dimension of the system (21) increases considerably, it is a significant problem to calculate the inverse of matrix $\mathbf{J}_s$, because it is very time consuming for a single computer to do all the calculations and meet the timing restrictions of the real-time simulation. In this case, it is beneficial to decompose the system into some local nodes, and the $\mathbf{J}_s$ matrix of each local node, which has a lower dimension compared with the overall system, is calculated on a separate computer. In order to decompose the overall system, the definition 3 and proposition 1 are applied.

Definition 3. The set of boundary algebraic equations is the set of algebraic equations that by eliminating them, the set of DAE subsystems (21) would be decomposed into multiple DAE local nodes which are independent from each other. Each local node is the combination of several subsystems and there are no two local nodes with common subsystems. Hence, each local node is a new subsystem in a new notation.

Example 1. The schematic of a set of boundary algebraic equations is shown in figure 3.

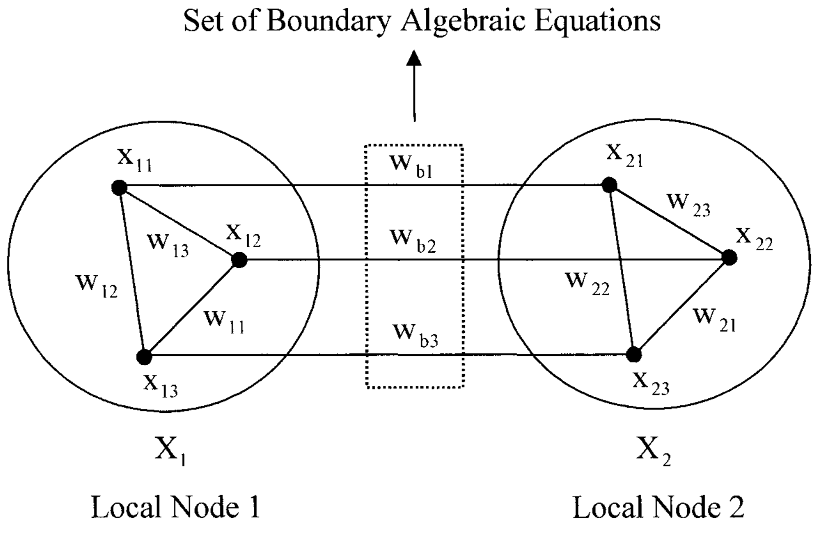


Figure 3. The schematic of a set of boundary algebraic equations.

In this figure, x_{ij} and w_{ij}'s are the states and algebraic equations, respectively. The set of boundary algebraic equations $W_b = \{w_{b1}, w_{b2}, w_{b3}\}$ decomposes the whole system into the two local nodes $X_1 = [x_{11} \quad x_{12} \quad x_{13}]^T$ and $X_2 = [x_{21} \quad x_{22} \quad x_{23}]^T$, with the corresponding algebraic equations $W_1 = [w_{11} \quad w_{12} \quad w_{13}]^T$ and $W_2 = [w_{21} \quad w_{22} \quad w_{23}]^T$, respectively.

2.1.2 Decoupling the DAE System via the Set of Boundary Algebraic Equations

The following definition introduces some new concepts.

Definition 4. The definition of boundary algebraic equations, classifies the inputs of the system into local inputs and boundary inputs. A local input $\overline{\mathbf{v}}_{\mathrm{i}}$ is the one which exclusively emerges in one local node (and no more local nodes), and a boundary input $\mathbf{v}_{\mathrm{bj}}$, corresponding to the boundary algebraic equation $\mathbf{w}_{\mathrm{bj}}$, is the one which emerges in more than one local node. If the boundary algebraic equation $\mathbf{w}_{\mathrm{bj}}$ connects k subsystems S_1, S_2, ..., S_k (and no other subsystems), and the boundary input $\mathbf{v}_{\mathrm{bj}}$ emerges in all the same k subsystems (and no other subsystems), then the boundary input $\mathbf{v}_{\mathrm{bj}}$ is the corresponding boundary input for the boundary algebraic equation $\mathbf{w}_{\mathrm{bj}}$. Hence, the boundary input and boundary algebraic equations could be defined as a boundary node $(\mathbf{v}_{\mathrm{bj}}, \mathbf{w}_{\mathrm{bj}})$.

Assumption 2. In this work, the concern is with the problems in which there exist boundary nodes. It means that there always exists a boundary input $\mathbf{v}_{\mathrm{bj}}$ corresponding to each boundary algebraic equation $\mathbf{w}_{\mathrm{bj}}$. This is a basic assumption for solvability.

Proposition 1. Considering assumption 2 for the set of DAE subsystems (21), it is possible to define the set of boundary algebraic equations $W_b = \{w_{b1}, w_{b2}, \ldots, w_{bL}\}$ including L algebraic equations, so that the whole system could be decomposed into multiple (N) local nodes of the following form:

$$\dot{\mathbf{x}}_i = \mathbf{f}_i(\mathbf{x}_i) + \mathbf{h}_i(\mathbf{x}_i, \mathbf{z}_i) + \sum_{j=1}^{L} \mathbf{h}_{ij}(\mathbf{x}, \mathbf{z}_{bj})$$

$$\dot{\mathbf{z}}_i = \mathbf{v}_i \qquad (i = 1, \ldots, N) \tag{25}$$

$$\mathbf{w}_i = \mathbf{g}_i(\mathbf{x}_i)$$

where $x_i \in R^{n'_i}$, $\mathbf{v}_i \in R^{m'_i}$ and $\mathbf{w}_i \in R^{m'_i}$ are the state, input, and output vectors of the i-th local node, respectively. The vector $\mathbf{x}$ is defined as $\mathbf{x} = \begin{bmatrix} \mathbf{x}_1^T & \mathbf{x}_2^T & \cdots & \mathbf{x}_N^T \end{bmatrix}^T$. For the boundary nodes, instead of one integrator, two integrators are applied to the corresponding boundary input $\mathbf{v}_{bi}$ to make the parameter $\mathbf{z}_{bi}$:

$$\ddot{\mathbf{z}}_{bj} = \mathbf{v}_{bj} \qquad (j = 1, \ldots, L)$$

$$\mathbf{w}_{bj} = \mathbf{g}_{bj}(\mathbf{x}) \tag{26}$$

where $\mathbf{v}_{bj} \in R^{m_{bj}}$ and $\mathbf{w}_{bj} \in R^{m_{bj}}$ are the input and output vectors of the j-th boundary node. The set of equations {(25),(26)} is the distributed realization of the DAE system (21).

Remark 1. Note that according to definition 3, eliminating the terms related to the boundary nodes $(\mathbf{v}_{bj}, \mathbf{w}_{bj})$, i.e. $\sum_{j=1}^{L} \mathbf{h}_{ij}(\mathbf{x}, \mathbf{z}_{bj})$, changes the local node (25) into the independent DAE local node:

$$\dot{\mathbf{x}}_i = \mathbf{f}_i(\mathbf{x}_i) + \mathbf{h}_i(\mathbf{x}_i, \mathbf{z}_i)$$

$$\dot{\mathbf{z}}_i = \mathbf{v}_i \qquad (i = 1, \ldots, N) \tag{27}$$

$$\mathbf{w}_i = \mathbf{g}_i(\mathbf{x}_i)$$

Example 2. To clarify the formulation above, a triple pendulum is considered in figure 4.

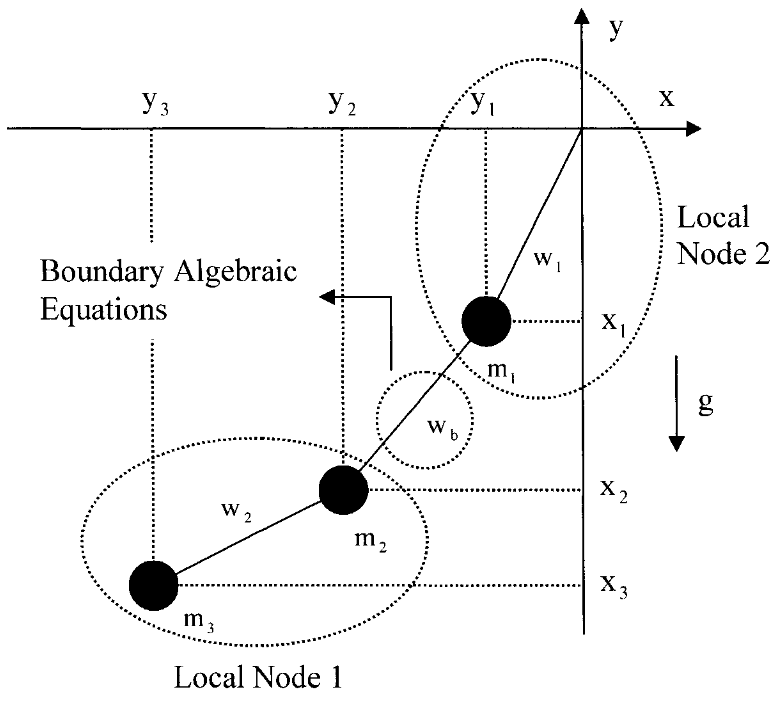


Figure 4. The triple pendulum problem and the distributed realization.

The dynamics equations are:

$$
\begin{aligned}
\ddot{x}_1 &= \frac{1}{m_1}[z_1(-x_1) - z_b(x_1 - x_2)] \\
\ddot{y}_1 &= \frac{1}{m_1}[z_1(-y_1) - z_b(y_1 - y_2)] - g \\
\ddot{x}_2 &= \frac{1}{m_2}[z_b(x_1 - x_2) - z_2(x_2 - x_3)] \\
\ddot{y}_2 &= \frac{1}{m_2}[z_b(y_1 - y_2) - z_2(y_2 - y_3)] - g \\
\ddot{x}_3 &= \frac{1}{m_3}[z_2(x_2 - x_3)] \\
\ddot{y}_3 &= \frac{1}{m_3}[z_2(y_2 - y_3)] - g \qquad (28) \\
\dot{z}_1 &= v_1 \\
\dot{z}_b &= v_b \\
\dot{z}_2 &= v_2 \\
w_1 &= x_1^2 + y_1^2 - l_1^2 \\
w_b &= (x_2 - x_1)^2 + (y_2 - y_1)^2 - l_b^2 \\
w_2 &= (x_3 - x_2)^2 + (y_3 - y_2)^2 - l_2^2
\end{aligned}
$$

The outputs of the system are the three lengths of the pendulums, which are desired to converge to l_1, l_b and l_2, respectively. The distributed model of the system includes the two local nodes:

$$\ddot{x}_1 = \frac{1}{m_1}[z_1(-x_1) - z_b(x_1 - x_2)]$$

$$\ddot{y}_1 = \frac{1}{m_1}[z_1(-y_1) - z_b(y_1 - y_2)] - g \tag{29}$$

$$\dot{z}_1 = v_1$$

$$w_1 = x_1^2 + y_1^2 - l_1^2$$

and

$$\ddot{x}_2 = \frac{1}{m_2}[z_b(x_1 - x_2) - z_2(x_2 - x_3)]$$

$$\ddot{y}_2 = \frac{1}{m_2}[z_b(y_1 - y_2) - z_2(y_2 - y_3)] - g$$

$$\ddot{x}_3 = \frac{1}{m_3}[z_2(x_2 - x_3)] \tag{30}$$

$$\ddot{y}_3 = \frac{1}{m_3}[z_2(y_2 - y_3)] - g$$

$$\dot{z}_2 = v_2$$

$$w_2 = (x_3 - x_2)^2 + (y_3 - y_2)^2 - l_2^2$$

And the boundary node (including the boundary algebraic equation with an extra integrator in the corresponding boundary input) is:

$$\ddot{z}_b = v_b$$

$$w_b = (x_2 - x_1)^2 + (y_2 - y_1)^2 - l_b^2 \tag{31}$$

The equations (29) and (30) represent two distributed local nodes of the main system, and the equation (31) represents the boundary node. Finally, {(29),(30),(31)} represents the distributed formulation of the triple pendulum system. First, the control laws v_1 and v_2 can be calculated on two separate distributed computers. Next, the control law v_b can be calculated on the central computer as well. Now, calculating the inverse transform of $\bar{\mathbf{J}}_{s1}$, $\bar{\mathbf{J}}_{s2}$ and $\bar{\mathbf{J}}_{sb}$ on three separate computers (distributed method, figure 5) is considerably less time consuming than calculating the $\mathbf{J}_s$ matrix of the overall system on a single computer (figure 1). To achieve a good performance in the distributed method, decentralized sliding mode control is applied.

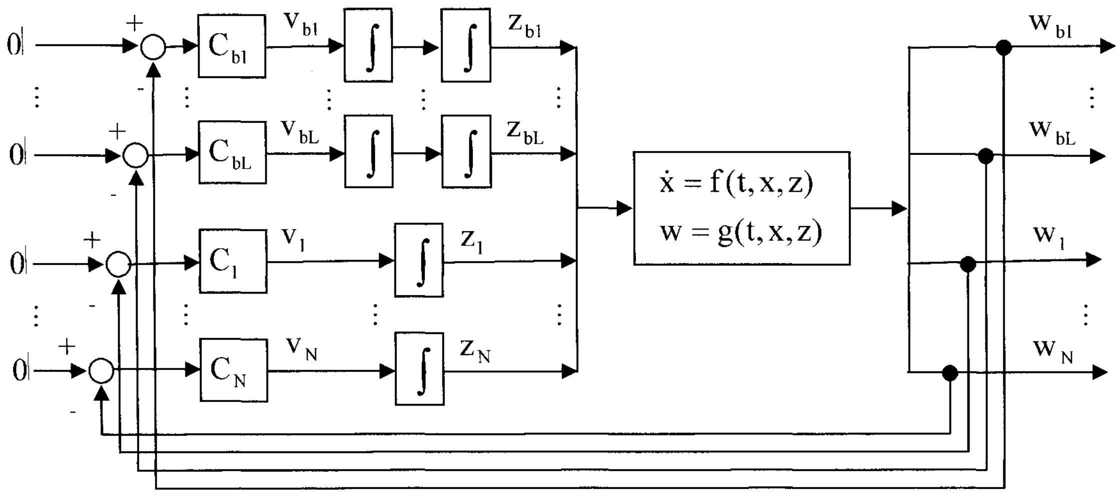


Figure 5. Distributed state space realization of a DAE.

2.2 Decentralized Sliding Mode Control

In order to control the distributed formulation of the system, decentralized sliding control approach is applicable. The local node (25) is transformed into the standard state space realization (32), where $\mathbf{x} = \begin{bmatrix} \mathbf{x}_1^T & \mathbf{x}_2^T & \cdots & \mathbf{x}_N^T \end{bmatrix}^T$.

$$\frac{d}{dt}\begin{bmatrix} \mathbf{x}_i \\ \mathbf{z}_i \end{bmatrix} = \begin{bmatrix} \mathbf{f}_i(\mathbf{x}_i) + \mathbf{h}_i(\mathbf{x}_i, \mathbf{z}_i) \\ \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix} \cdot \mathbf{v}_i + \begin{bmatrix} \sum_{j=1}^{L} \mathbf{h}_{ij}(\mathbf{x}, \mathbf{z}_{bj}) \\ \mathbf{0} \end{bmatrix} \tag{32}$$

$$\mathbf{w}_i = \mathbf{g}_i(\mathbf{x}_i) \qquad\qquad (i = 1, \ldots, N)$$

And the set of boundary nodes (26), accompanied by the corresponding dynamics inherited from all local nodes (25), can be reformulated accordingly as:

$$\frac{d}{dt}\begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_N \\ \mathbf{z}_1 \\ \vdots \\ \mathbf{z}_N \\ \mathbf{z}_{b1} \\ \vdots \\ \mathbf{z}_{bL} \\ \dot{\mathbf{z}}_{b1} \\ \vdots \\ \dot{\mathbf{z}}_{bL} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1(\mathbf{x}_1) + \mathbf{h}_1(\mathbf{x}_1, \mathbf{z}_1) + \mathbf{h}_{11}(\mathbf{x}, \mathbf{z}_{b1}) + \cdots \mathbf{h}_{1L}(\mathbf{x}, \mathbf{z}_{bL}) \\ \vdots \\ \mathbf{f}_N(\mathbf{x}_N) + \mathbf{h}_N(\mathbf{x}_N, \mathbf{z}_N) + \mathbf{h}_{N1}(\mathbf{x}, \mathbf{z}_{b1}) + \cdots \mathbf{h}_{NL}(\mathbf{x}, \mathbf{z}_{bL}) \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \dot{\mathbf{z}}_{b1} \\ \vdots \\ \dot{\mathbf{z}}_{bL} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \\ \mathbf{I} \\ \\ \end{bmatrix} \cdot \begin{bmatrix} \mathbf{v}_{b1} \\ \vdots \\ \mathbf{v}_{bL} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \\ \mathbf{I} \\ \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_N \end{bmatrix} \tag{33}$$

$$\mathbf{w}_b = \mathbf{g}_b(\mathbf{x})$$

where $\mathbf{x} = \begin{bmatrix}\mathbf{x}_1^T & \mathbf{x}_2^T & \cdots & \mathbf{x}_N^T\end{bmatrix}^T$, $\mathbf{w}_b = \begin{bmatrix}\mathbf{w}_{b1}^T & \mathbf{w}_{b2}^T & \cdots & \mathbf{w}_{bL}^T\end{bmatrix}^T$ and $\mathbf{g}_b(\mathbf{x}) = \begin{bmatrix}\mathbf{g}_{b1}^T(\mathbf{x}) & \mathbf{g}_{b2}^T(\mathbf{x}) & \cdots & \mathbf{g}_{bL}^T(\mathbf{x})\end{bmatrix}^T$. Now, these standard state space formulations (32) and (33) are considered in the following section.

2.2.1 Output Linearization

Consider the local node (32) in the general form:

$$\text{Local Node } P_i \ (i=1,\dots,N): \begin{cases} \dot{\bar{\mathbf{x}}}_i = \bar{\mathbf{f}}_i(\bar{\mathbf{x}}_i) + \bar{\mathbf{h}}_i(\bar{\mathbf{x}}_i)\mathbf{v}_i + \boldsymbol{\theta}_i(\bar{\mathbf{x}}) \\ \mathbf{w}_i = \bar{\mathbf{g}}_i(\bar{\mathbf{x}}_i) \end{cases} \tag{34}$$

where $\bar{\mathrm{x}}_i = \begin{bmatrix}\mathbf{x}_i & \mathbf{z}_i\end{bmatrix}^T \in R^{N_i}$, $\mathbf{v}_i \in R^{M_i}$ and $\mathbf{w}_i \in R^{M_i}$ are the state, input and output vectors of the i-th local node, respectively. Also, $\bar{\mathbf{f}}_i$, $\bar{\mathbf{h}}_i$ and $\bar{\mathbf{g}}_i$ are the vector functions with appropriate dimensions. $\boldsymbol{\theta}_i$ is the interconnection vector including the term $\sum_{j=1}^{L}\mathbf{h}_{ij}(\mathbf{x}_i,\mathbf{z}_{bj})$. This vector has a well defined dimension. Since we have access to these interconnection term (by having access to $\mathbf{x}$, states of other local nodes, and $\mathbf{z}_{bj}$, states of boundary nodes), we are willing to approximately calculate this term.

For the boundary nodes (33) the general form is:

$$\text{Set of Boundary Nodes}: \begin{cases} \dot{\bar{\mathbf{x}}}_b = \bar{\mathbf{f}}_b(\bar{\mathbf{x}}_b) + \bar{\mathbf{h}}_b(\bar{\mathbf{x}}_b)\mathbf{v}_b + \mathbf{d}_b(\mathbf{v}) \\ \mathbf{w}_b = \bar{\mathbf{g}}_b(\bar{\mathbf{x}}_b) \end{cases} \tag{35}$$

where $\bar{\mathbf{x}}_b = [\mathbf{x}_1^T \;\cdots\; \mathbf{x}_N^T \;\; \mathbf{z}_1^T \;\cdots\; \mathbf{z}_N^T \;\; \mathbf{z}_{b1}^T \;\cdots\; \mathbf{z}_{bL}^T \;\; \dot{\mathbf{z}}_{b1}^T \;\cdots\; \dot{\mathbf{z}}_{bL}^T]^T \in R^{N_b}$,

$\mathbf{v}_b = [\mathbf{v}_{b1}^T \;\cdots\; \mathbf{v}_{bL}^T]^T \in R^{M_b}$, and $\mathbf{w}_b = [\mathbf{w}_{b1}^T \;\cdots\; \mathbf{w}_{bL}^T]^T \in R^{M_b}$ $(M_b = \sum_{j=1}^{L} M_{bj}$, $N_b = \sum_{i=1}^{N} N_i + 2M_b)$.

The vector $\mathbf{d}_b(\mathbf{v})$ represents the system uncertainties and external disturbances. It includes the effect of the local inputs $\mathbf{v}_i$ (i=1,2,...,N). Since we are concerned with decentralized control and distributed computation, we are willing to estimate $\mathbf{d}_b(\mathbf{v})$ instead of approximately calculating this term. This estimation is done by applying adaptive updating law to the coefficients of Fourier series expansion. Obviously, the set of boundary nodes (35) has a similar formulation structure to each of the local nodes P_i in (34).

Remark 2. To avoid further confusion, the notation $\underline{L}$ represents the Lie derivative operator for the vector functions $\mathbf{f}(\mathbf{x})$ and $\mathbf{g}(\mathbf{x})$ as follow:

$$\underline{\mathbf{L}}_{\mathbf{f}(\mathbf{x})}\mathbf{g}(\mathbf{x}) = \nabla\mathbf{g}(\mathbf{x})\cdot\mathbf{f}(\mathbf{x}) \tag{36}$$

Remark 3. To avoid further confusion, the notation $\underline{\underline{L}}$ represents Lie derivative operator for the matrix function $\mathbf{h}(\mathbf{x})$ and the vector function $\mathbf{g}(\mathbf{x})$ as follow:

$$\underline{\underline{\mathbf{L}}}_{\mathbf{h}(\mathbf{x})}\mathbf{g}(\mathbf{x}) = \nabla\mathbf{g}(\mathbf{x})\cdot\mathbf{h}(\mathbf{x}) \tag{37}$$

Proposition 2. Consider the general formulation (38), which is similar to both (34) and (35) with a minor difference that will be expressed ahead in remarks 4 and 5, respectively [38].

$$\begin{cases} \dot{\bar{\mathbf{x}}}_i = \bar{\mathbf{f}}_i(\bar{\mathbf{x}}_i) + \bar{\mathbf{h}}_i(\bar{\mathbf{x}}_i)\mathbf{v}_i + \boldsymbol{\theta}_i(\bar{\mathbf{x}}) + \mathbf{d}_i(.) \\ \mathbf{w}_i = \bar{\mathbf{g}}_i(\bar{\mathbf{x}}_i) \end{cases} \quad (i = 1,2,\ldots,N,b) \tag{38}$$

$\bar{x}_i \in R^{N_i}$, $\mathbf{v}_i \in R^{M_i}$ and $\mathbf{w}_i \in R^{M_i}$ are the state, input and output vectors of the i-th node, respectively. Considering assumption 1, suppose that this node has a vector index with equal elements $r_i = 3$ at $\mathbf{x}_{i0}$ and $R_i = M_i r_i \le N_i$. Set:

$$\begin{aligned} &\mathbf{y}_1 = \bar{\mathbf{g}}(\bar{\mathbf{x}}) \\ &\vdots \\ &\mathbf{y}_{ir_i} = \mathbf{L}_{\bar{\mathbf{f}}_i}^{r_i - 1}\bar{\mathbf{g}}_i(\bar{\mathbf{x}}_i) \end{aligned} \tag{39}$$

If $R_i = M_i \cdot r_i$ is strictly less than N_i, it is always possible to find $N_i - R_i$ more functions $\eta_{i(R_i+1)},\ldots,\eta_{iN_i}$ such that the mapping

$$\begin{bmatrix} \mathbf{y}_{i1}^T & \cdots & \mathbf{y}_{ir_i}^T & \eta_{i(R_i+1)} & \cdots & \eta_{iN_i} \end{bmatrix}^T \tag{40}$$

has a Jacobean matrix that is nonsingular at $\mathbf{x}_{i0}$. The $\eta_{i(R_i+1)},\ldots,\eta_{iN_i}$ are chosen to satisfy

$$\mathbf{L}_{\bar{\mathbf{h}}_i}\eta_{ij} = 0 \quad , \quad 1 \le i \le N \quad , \quad R_i + 1 \le j \le N_i \tag{41}$$

Now, set

$$[\mathbf{y}_i]_{R_i\times 1} = [\mathbf{y}_{i1}^T \quad \cdots \quad \mathbf{y}_{ir_i}^T]^T \quad , \quad [\boldsymbol{\eta}_i]_{(N_i-R_i)\times 1} = [\eta_{i(R_i+1)} \quad \cdots \quad \eta_{iN_i}]^T \tag{42}$$

Assuming that the state coordinate transformation $(\mathbf{y}_i, \boldsymbol{\eta}_i) = \mathbf{T}_i(\mathbf{x}_i)$ is performed to the node P_i in (38), the following form can be obtained:

$$\begin{aligned}
\dot{\mathbf{y}}_{i1} &= \mathbf{y}_{i2} + \Delta\boldsymbol{\theta}_{i1}(\mathbf{y},\boldsymbol{\eta}) + \Delta\mathbf{d}_{i1}(\mathbf{y}_i,\boldsymbol{\eta}_i,\mathbf{d}_i) \\
&\vdots \\
\dot{\mathbf{y}}_{ir_i} &= \boldsymbol{\alpha}_i(\mathbf{y}_i,\boldsymbol{\eta}_i) + \boldsymbol{\beta}_i(\mathbf{y}_i,\boldsymbol{\eta}_i)\mathbf{v}_i + \Delta\boldsymbol{\theta}_{ir_i}(\mathbf{y},\boldsymbol{\eta}) + \Delta\mathbf{d}_{ir_i}(\mathbf{y}_i,\boldsymbol{\eta}_i,\mathbf{d}_i) \\
\dot{\boldsymbol{\eta}}_i &= \mathbf{q}_i(\mathbf{y}_i,\boldsymbol{\eta}_i) + \Delta\mathbf{Q}_i(\mathbf{y},\boldsymbol{\eta},\mathbf{d}_i)
\end{aligned} \tag{43}$$

Where

$$\begin{aligned}
&\boldsymbol{\alpha}_i(\mathbf{y}_i,\boldsymbol{\eta}_i) = \underline{\mathbf{L}}_{\underline{\mathbf{f}}_i}^{r_i}\bar{\mathbf{g}}_i \circ \mathbf{T}_i^{-1}(\mathbf{y}_i,\boldsymbol{\eta}_i) && i = 1,\ldots,N \\
&\boldsymbol{\beta}_i(\mathbf{y}_i,\boldsymbol{\eta}_i) = \underline{\mathbf{L}}_{\underline{\bar{\mathbf{h}}}_i}\underline{\mathbf{L}}_{\underline{\mathbf{f}}_i}^{r_i-1}\bar{\mathbf{g}}_i \circ \mathbf{T}_i^{-1}(\mathbf{y}_i,\boldsymbol{\eta}_i) && j = 1,\ldots,r_i \\
&\Delta\boldsymbol{\theta}_{ij}(\mathbf{y},\boldsymbol{\eta}) = \underline{\mathbf{L}}_{\theta_i}\underline{\mathbf{L}}_{\underline{\mathbf{f}}_i}^{j-1}\bar{\mathbf{g}}_i(\mathbf{x}_i) \circ \mathbf{T}_i^{-1}(\mathbf{y}_i,\boldsymbol{\eta}_i) \\
&\Delta\mathbf{d}_{ij}(\mathbf{y}_i,\boldsymbol{\eta}_i,\mathbf{d}_i) = \underline{\mathbf{L}}_{\mathbf{d}_i}\mathbf{L}_{\underline{\mathbf{f}}_i}^{j-1}\bar{\mathbf{g}}_i(\mathbf{x}_i) \circ \mathbf{T}_i^{-1}(\mathbf{y}_i,\boldsymbol{\eta}_i) \\
&\Delta\mathbf{Q}_i(\mathbf{y},\boldsymbol{\eta},\mathbf{d}_i) = [\underline{\mathbf{L}}_{\theta_i}\boldsymbol{\eta}_{i(R_i+1)} + \underline{\mathbf{L}}_{\mathbf{d}_i}\boldsymbol{\eta}_{i(R_i+1)} \quad \cdots \quad \underline{\mathbf{L}}_{\theta_i}\eta_{iN_i} + \underline{\mathbf{L}}_{\mathbf{d}_i}\boldsymbol{\eta}_{iN_i}]^T \circ \mathbf{T}_i^{-1}(\mathbf{y}_i,\boldsymbol{\eta}_i) \\
&\mathbf{y} = [\mathbf{y}_1^T \quad \cdots \quad \mathbf{y}_N^T]^T \\
&\boldsymbol{\eta} = [\boldsymbol{\eta}_1^T \quad \cdots \quad \boldsymbol{\eta}_N^T]^T
\end{aligned} \tag{44}$$

Therefore, the node P_i in (43) can be expressed as:

$$\begin{aligned}
\mathbf{w}_i &= \bar{\mathbf{g}}_i = \mathbf{y}_{i1} \\
\mathbf{w}_i^{(1)} &= \dot{\mathbf{y}}_{i1} = \mathbf{y}_{i2} + \Delta\boldsymbol{\theta}_{i1} + \Delta\mathbf{d}_{i1} \\
&\vdots \\
\mathbf{w}_i^{(r_i)} &= \dot{\mathbf{y}}_{ir_i} = \boldsymbol{\alpha}_i + \boldsymbol{\beta}_i\mathbf{v}_i + \Delta\boldsymbol{\Theta}_i + \Delta\mathbf{D}_i
\end{aligned} \tag{45}$$

where

$$\Delta\boldsymbol{\Theta}_i = \Delta\boldsymbol{\theta}_{i1}^{(r_i-1)} + \Delta\boldsymbol{\theta}_{i2}^{(r_i-2)} + \cdots + \Delta\boldsymbol{\theta}_{i(r_i-1)}^{(1)} + \Delta\boldsymbol{\theta}_{ir_i} \tag{46}$$
$$\Delta\mathbf{D}_i = \Delta\mathbf{d}_{i1}^{(r_i-1)} + \Delta\mathbf{d}_{i2}^{(r_i-2)} + \cdots + \Delta\mathbf{d}_{i(r_i-1)}^{(1)} + \Delta\mathbf{d}_{ir_i}$$

Remark 4. Applying proposition 2 to the local node (34) results in $\Delta\mathbf{D}_i = \mathbf{0}$ and $\Delta\boldsymbol{\Theta}_i \neq \mathbf{0}$ which contains the interconnection terms (but no boundary input, since $r_b = r_i + 1$).

Remark 5. Applying proposition 2 to the set of boundary nodes (35) results in $\Delta\boldsymbol{\Theta}_b = \mathbf{0}$ and $\Delta\mathbf{D}_b \neq \mathbf{0}$ which is in terms of local inputs and their first derivatives (since $r_i = r_b - 1$).

Assumption 3. The zero dynamics $\dot{\boldsymbol{\eta}}_i = \mathbf{q}_i(\mathbf{0}, \boldsymbol{\eta}_i)$ is exponentially stable in the domain of definition.

In order to estimate some of the complicated terms including interconnections, disturbances and modelling uncertainties, it is possible to use the Fourier series approximation of those terms [36], and apply some adaptive laws [7,14,35] in order to adjust the corresponding Fourier coefficients.

2.2.2 Fourier series Expansion

Proposition 3. Consider ΔD_{bj}, the j-th element of the vector $\Delta\mathbf{D}_b$, as a function of time $\Delta D_{bj}(t)$. If $\Delta D_{bj}(t)$ satisfies the Dirichlet conditions (a piecewise regular function that has a finite number of finite discontinuities and has a finite number of extrema) [36], then it can be expanded in a Fourier series within a time interval [0,T] as:

$$\Delta D_{bj}(t) = a_{j0} + \sum_{k=1}^{\infty}\left(a_{jk}\cos\left(\frac{2k\pi}{T}t\right) + b_{jk}\sin\left(\frac{2k\pi}{T}t\right)\right) \tag{47}$$

The approximation is

$$\Delta D_{bj}(t) \cong a_{j0} + \sum_{k=1}^{n_c}\left(a_{jk}\cos\left(\frac{2k\pi}{T}t\right) + b_{jk}\sin\left(\frac{2k\pi}{T}t\right)\right) \tag{48}$$

and the approximation error is

$$e(t) = \sum_{k=n_c+1}^{\infty}\left(a_{jk}\cos\left(\frac{2k\pi}{T}t\right) + b_{jk}\sin\left(\frac{2k\pi}{T}t\right)\right) \le \sum_{k=n_c+1}^{\infty}\left(|a_{jk}| + |b_{jk}|\right) \tag{49}$$

Remark 6. Equation (47) is the Fourier series expansion of a periodic function. Since ΔD_{bj} is an aperiodic function, choosing a sufficiently large T makes equation (48) a proper approximation of the function.

Remark 7. The advantage of equation (48) is the linear parameterization of the function $\Delta D_{bj}(t)$ into a basis function vector $\mathbf{F}(t)$ and a coefficient vector ψ_{bj} as below:

$$\Delta D_{bj}(t) \cong \psi_{bj}^{T} \cdot \mathbf{F}(t) \tag{50}$$

where

$$\psi_{bj} = \begin{bmatrix} a_{j0} & a_{j1} & b_{j1} & \cdots & a_{jn_\varepsilon} & a_{jn_\varepsilon} \end{bmatrix}^T$$
$$\mathbf{F}(t) = \begin{bmatrix} 1 & \cos\left(\frac{2\pi}{T}t\right) & \sin\left(\frac{2\pi}{T}t\right) & \cdots & \cos\left(\frac{2n_\varepsilon\pi}{T}t\right) & \sin\left(\frac{2n_\varepsilon\pi}{T}t\right) \end{bmatrix}^T \tag{51}$$

Therefore, for the whole matrix function $\Delta\mathbf{H}_b$, the Fourier series expansion approximation is:

$$\Delta\mathbf{D}_b(t) \cong \mathbf{\Psi}_b^T \cdot \mathbf{F}(t) \tag{52}$$

where

$$\mathbf{\Psi}_b = \begin{bmatrix} \psi_{b1} & \psi_{b2} & \cdots & \psi_{bM_i} \end{bmatrix} \tag{53}$$

Later on, adaptive updating laws are applied to estimate the coefficient matrix $\mathbf{\Psi}_i$ above [7,14,35].

2.2.3 Decentralized Sliding Mode Control

In general, sliding mode control is a robust method to control the systems with disturbances, uncertainties, and unmodelled dynamics. The simplicity of designing a sliding controller makes it the best candidate for controlling purposes. In all the designing procedures, we are concerned with designing the sliding gain k. Suppose that s is the sliding surface corresponding to the control output. Then $\dot{s}$, derivative of the sliding surface, is as follows:

$$\dot{s} = \alpha + J_s \nu \tag{54}$$

Therefore, choosing the controller

$$\nu = J_s^{-1}\left(-k\,\mathrm{sgn}(s)\right) \tag{55}$$

such that

$$k > \max(\alpha) \tag{56}$$

stabilizes the closed-loop system. However the term $\max(\alpha)$ could probably be a large number, and so k should be chosen sufficiently large to cancel the effect of α. This large k makes the dynamic of the closed-loop system stiff. As a result, for simulation of a stiff dynamic system, it is necessary to choose the simulation step size small enough to make the equivalent discrete model of the system stable. This small step size is not a favourable choice, since in most applications there are some timing restrictions involved with either the real-time simulation constraints or the processor speed of the computer on which the simulation is being done. In such cases, it is beneficial to estimate the term α by $\hat{\alpha}$ so that the gain k is not responsible to cancel the effect of α any more, but only responsible to cancel the effect of the estimation error $\alpha - \hat{\alpha}$.

$$k > \max(\alpha - \hat{\alpha}) \tag{57}$$

The choice of estimation is either a neuro-fuzzy network or Fourier series estimation. The later will be used in the next section. In this case, the estimation may not be a precise and exact one, but as long as it makes the amplitude of the error signal comparably less than the amplitude of the main signal, it is beneficial. It will be shown later (in example 5) how estimation decreases the order of sliding gain k.

In this chapter, the definition of a set of boundary algebraic equations was introduced. This concept was used to break the original DAE system and divide it into multiple nodes that are separate and independent from each other. Decentralized sliding mode control was applied to the distributed realization of the system. The interconnection and disturbance terms were approximated by their Fourier series expansion, and the Fourier coefficients adjusted by the corresponding adaptation laws. In the next chapter the design of suitable decentralizaed sliding controllers is investigated.

3 . Designing the Decentralized Sliding Controllers for the Decoupled DAE

In this section, the decentralized controllers are designed for the local nodes and the boundary node of the DAE system, separately. Therefore, the overall decentralized controller will stabilize the overall DAE system. The advantage of adaptive Fourier series approximation is explained, and a flow chart is presented in order to select the parameters of the decentralized sliding mode controller.

3.1 The Main Theorem

According to remark 4, proposition 2 reformulates local node (34) into the form (45), while the term $\Delta\mathbf{D}_i$ is equal to zero. Moreover, according to remark 5, proposition 2 reformulates the set of boundary nodes (35) into the form (45), while the term $\Delta\boldsymbol{\Theta}_b$ is equal to zero. The following theorem is applied [16,17,20].

Theorem 3. Considering the stability of internal dynamics (assumption 3), in order to simulate the distributed realization {(25),(26)} in the proposition 1, using the reformulation {(34),(35)},

a) First, the decentralized sliding controllers are designed on the distributed computers to derive the local inputs $\mathbf{v}_i$ (i=1,...,N) for the N local nodes (34);
b) Next, the decentralized sliding controller is designed on the distributed computers to derive the boundary inputs $\mathbf{v}_b$ for the set of L boundary nodes (35);

And the overall closed-loop distributed system will be stable. From physical considerations it can be shown that all the states of the distributed model are bounded near the solution.

Proof.

Considering assumption 1, suppose that the local node (34) has a vector index with equal elements r_i, and the internal dynamics are stable, and consider the sliding manifold:

$$\mathbf{s}_i = \sum_{j=1}^{r_i}\left(\binom{r_i - 1}{j-1}\mu_i^{r_i - j}\mathbf{w}_i^{(r_i - j)}\right) \tag{58}$$

The dynamic characteristic equation of the manifold is:

$$\sum_{j=1}^{r_i}\left(\binom{r_i - 1}{j-1}\mu_i^{r_i - j}p^{(r_i - j)}\right) = 0 \tag{59}$$

It has $r_i - 1$ negative poles at $\dfrac{-1}{\mu_i}$ $(\mu_i > 0)$. Using equations (45) the derivatives of the manifold is:

$$\dot{\mathbf{s}}_i = \sum_{j=2}^{r_i}\left(\binom{r_i-1}{j-1}\mu_i^{r_i-j}\mathbf{w}_i^{(r_i-j+1)}\right) + \mu_i^{r_i-1}\mathbf{w}_i^{(r_i)}$$
$$= \sum_{j=2}^{r_i}\left(\binom{r_i-1}{j-1}\mu_i^{r_i-j}\mathbf{w}_i^{(r_i-j+1)}\right) + \mu_i^{r_i-1}(\boldsymbol{\alpha}_i + \boldsymbol{\beta}_i\mathbf{v}_i + \Delta\boldsymbol{\Theta}_i) \qquad (60)$$

According to remark 4, $\Delta\boldsymbol{\Theta}_i$ represents the effect of the interconnections of the i-th local node. Assuming the term $\hat{\boldsymbol{\alpha}}_i$ is the approximation of the term $\boldsymbol{\alpha}_i$ such that $|\hat{\boldsymbol{\alpha}}_i - \boldsymbol{\alpha}_i| < \mathbf{E}_{\alpha_i}$, and that there exists an upper bound as $|\Delta\boldsymbol{\Theta}_i| < \mathbf{B}_{\Delta\Theta_i}$, the control law

$$\mathbf{v}_i = \bar{\mathbf{J}}_{si}^{-1}\left(-\sum_{j=2}^{r_i}\left(\binom{r_i-1}{j-1}\mu_i^{r_i-j}\mathbf{w}_i^{(r_i-j+1)}\right) - \mu_i^{r_i-1}\hat{\boldsymbol{\alpha}}_i - \mathrm{diag}(\mathbf{k}_i).\mathbf{sat}(\varepsilon_i^{-1}\mathrm{diag}(\boldsymbol{\delta}_i)\mathbf{s}_i)\right) \qquad (61)$$

with

$$\bar{\mathbf{J}}_{si} = \mu_i^{r_i-1}\boldsymbol{\beta}_i \qquad (62)$$

stabilizes the local node (34) if the following condition is satisfied:

$$\mathbf{k}_i \geq \mu_i^{r_i-1}\mathbf{E}_{\alpha_i} + \mu_i^{r_i-1}\mathbf{B}_{\Delta\Theta_i} + \boldsymbol{\gamma}_i \qquad (63)$$

where all elements of the vector $\boldsymbol{\gamma}_i$ are positive.

Considering assumption 1, suppose that the boundary node (35) has a vector index with equal elements $r_b = r_i + 1$ (because of the extra integrator), and consider the sliding manifold:

$$\mathbf{s}_b = \sum_{j=1}^{r_b}\left(\binom{r_b-1}{j-1}\mu_b^{r_b-j}\mathbf{w}_b^{(r_b-j)}\right) \tag{64}$$

The dynamic characteristic equation of the manifold is:

$$\sum_{j=1}^{r_b}\left(\binom{r_b-1}{j-1}\mu_b^{r_b-j}p^{(r_b-j)}\right) = 0 \tag{65}$$

It has $r_b - 1$ negative poles at $\frac{-1}{\mu_b}$ ($\mu_b > 0$). Using equations (45), the derivatives of the manifold is

$$\begin{aligned}\dot{\mathbf{s}}_b &= \sum_{j=2}^{r_b}\left(\binom{r_b-1}{j-1}\mu_b^{r_b-j}\mathbf{w}_b^{(r_b-j+1)}\right) + \mu_b^{r_b-1}\mathbf{w}_b^{(r_b)} \\ &= \sum_{j=2}^{r_b}\left(\binom{r_b-1}{j-1}\mu_b^{r_b-j}\mathbf{w}_b^{(r_b-j+1)}\right) + \mu_b^{r_b-1}\left(\boldsymbol{\alpha}_b + \boldsymbol{\beta}_b\mathbf{v}_b + \Delta\mathbf{D}_b\right)\end{aligned} \tag{66}$$

According to remark 5, $\Delta\mathbf{D}_b$ represents the effects of the local inputs, as well as their derivatives, on the set of boundary nodes. Assuming the term $\hat{\boldsymbol{\alpha}}_b$ is the approximation of the term $\boldsymbol{\alpha}_b$ such that $|\hat{\boldsymbol{\alpha}}_b - \boldsymbol{\alpha}_b| < \mathbf{E}_{\boldsymbol{\alpha}_b}$, and that there exists an upper bound as $|\Delta\mathbf{D}_b| < \mathbf{B}_{\Delta\mathbf{D}_b}$, the control law

$$\mathbf{v}_b = \mathbf{J}_{sb}^{-1}\left(-\sum_{j=2}^{r_b}\left(\binom{r_b-1}{j-1}\mu_b^{r_b-j}\mathbf{w}_b^{(r_b-j+1)}\right) - \mu_b^{r_b-1}\hat{\boldsymbol{\alpha}}_b - \mathrm{diag}(\mathbf{k}_b).\mathbf{sat}(\varepsilon_b^{-1}\mathrm{diag}(\boldsymbol{\delta}_b)\mathbf{s}_b)\right) \tag{67}$$

stabilizes the set of boundary nodes (35) if the following condition is satisfied:

$$\mathbf{k}_b \geq \mu_b^{r_b-1}\mathbf{E}_{\boldsymbol{\alpha}_b} + \mu_b^{r_b-1}\mathbf{B}_{\Delta\mathbf{D}_b} + \boldsymbol{\gamma}_b \tag{68}$$

where all elements of the vector $\boldsymbol{\gamma}_b$ are positive.

The control laws (61) and (67) designed so far will stabilize the distributed realization $\{(34),(35)\}$. Consider the quadratic Lyapunov function:

$$V_t = \frac{1}{2}\sum_{i=1}^{N}\mathbf{s}_i^T\mathbf{s}_i + \frac{1}{2}\mathbf{s}_b^T\mathbf{s}_b \geq 0 \tag{69}$$

Using the manifold derivatives (60) and (66), and applying the control inputs (61) and (67), the derivative of the Lyapunov function is:

$$\begin{aligned}\frac{dV_t}{dt} &= \sum_{i=1}^{N}\mathbf{s}_i^T\dot{\mathbf{s}}_i + \mathbf{s}_b^T\dot{\mathbf{s}}_b \\ &= \sum_{i=1}^{N}\mathbf{s}_i^T\left(\mu_i^{r_i-1}(\boldsymbol{\alpha}_i - \hat{\boldsymbol{\alpha}}_i) + \mu_i^{r_i-1}\Delta\boldsymbol{\Theta}_i - \text{diag}(\mathbf{k}_i)\mathbf{sat}(\varepsilon_i^{-1}\text{diag}(\boldsymbol{\delta}_i)\mathbf{s}_i)\right) \\ &\quad + \mathbf{s}_b^T\left(\mu_b^{r_b-1}(\boldsymbol{\alpha}_b - \hat{\boldsymbol{\alpha}}_b) + \mu_b^{r_b-1}\Delta\mathbf{D}_b - \text{diag}(\mathbf{k}_b)\mathbf{sat}(\varepsilon_b^{-1}\text{diag}(\boldsymbol{\delta}_b)\mathbf{s}_b)\right)\end{aligned} \tag{70}$$

The upper bounds $|\hat{\boldsymbol{\alpha}}_i - \boldsymbol{\alpha}_i| < \mathbf{E}_{\boldsymbol{\alpha}_i}$, $|\Delta\boldsymbol{\Theta}_i| < \mathbf{B}_{\Delta\boldsymbol{\Theta}_i}$, $|\hat{\boldsymbol{\alpha}}_b - \boldsymbol{\alpha}_b| < \mathbf{E}_{\Delta\boldsymbol{\Theta}_i}$ and $|\Delta\mathbf{D}_b| < \mathbf{B}_{\Delta\mathbf{D}_b}$ certify that:

$$\begin{aligned}\frac{dV_t}{dt} \leq &\sum_{i=1}^{N}\mathbf{s}_i^T\left(\mu_i^{r_i-1}\mathbf{E}_{\boldsymbol{\alpha}_i} + \mu_i^{r_i-1}\mathbf{B}_{\Delta\boldsymbol{\Theta}_i} - \text{diag}(\mathbf{k}_i)\mathbf{sat}(\varepsilon_i^{-1}\text{diag}(\boldsymbol{\delta}_i)\mathbf{s}_i)\right) \\ &+ \mathbf{s}_b^T\left(\mu_b^{r_b-1}\mathbf{E}_{\Delta\boldsymbol{\Theta}_b} + \mu_i^{r_i-1}\mathbf{B}_{\Delta\mathbf{D}_b} - \text{diag}(\mathbf{k}_b)\mathbf{sat}(\varepsilon_b^{-1}\text{diag}(\boldsymbol{\delta}_b)\mathbf{s}_b)\right)\end{aligned} \tag{71}$$

Now, considering the conditions (63) and (68) results in:

$$\frac{dV_t}{dt} \leq \sum_{i=1}^{N} \mathbf{s}_i^T \left(-\operatorname{diag}(\boldsymbol{\gamma}_i)\mathbf{sat}(\varepsilon_i^{-1}\operatorname{diag}(\boldsymbol{\delta}_i)\mathbf{s}_i)\right) + \mathbf{s}_b^T \left(-\operatorname{diag}(\boldsymbol{\gamma}_b)\mathbf{sat}(\varepsilon_b^{-1}\operatorname{diag}(\boldsymbol{\delta}_b)\mathbf{s}_b)\right) \tag{72}$$

Since all elements of the vectors $\boldsymbol{\gamma}_i$ and $\boldsymbol{\gamma}_b$ are positive, it is concluded that

$$\frac{dV_t}{dt} \leq 0 \tag{73}$$

So the stability of the closed-loop distributed realization is guaranteed.

□

Example 3 . As an example, the nonlinear model of the triple pendulum problem in example 2 is considered. To design a decentralized controller, theorem 3 is applied. The matrix $\bar{\mathbf{J}}_s$ of the two subsystems and the boundary algebraic equations are calculated on three separate computers:

$$\bar{J}_{s1} = \frac{-2\mu^2}{m_1}\left(x_1^2 + y_1^2\right)$$

$$\bar{J}_{sb} = -2\mu^2 \cdot \left(\frac{1}{m_2} + \frac{1}{m_1}\right)\left[(x_2 - x_1)^2 + (y_2 - y_1)^2\right]$$

$$\bar{J}_{s2} = -2\mu^2 \cdot \left(\frac{1}{m_3} + \frac{1}{m_2}\right)\left[(x_3 - x_2)^2 + (y_3 - y_2)^2\right]$$

The typical parameter values used in the simulation are as follow:

Masses	$m_1 = m_2 = m_3 = 1\text{Kg}$
Desired lengths	$l_1 = l_b = l_2 = 1\text{m}$
Sliding parameters	$\mu = 0.05$, $k = 50$
Simulation step size	$T = 0.001\text{Sec}$
Boundary layer thickness	$\varepsilon = 0.1$

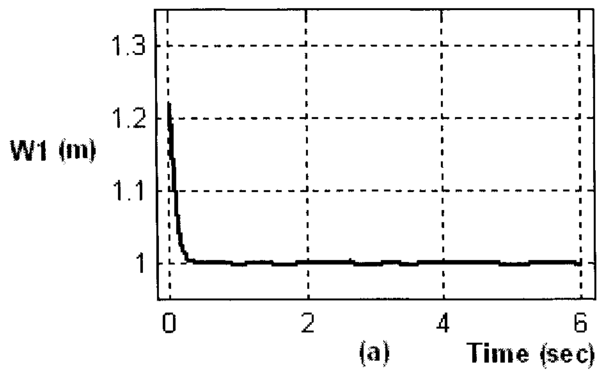
Assuming the state vector of the system is

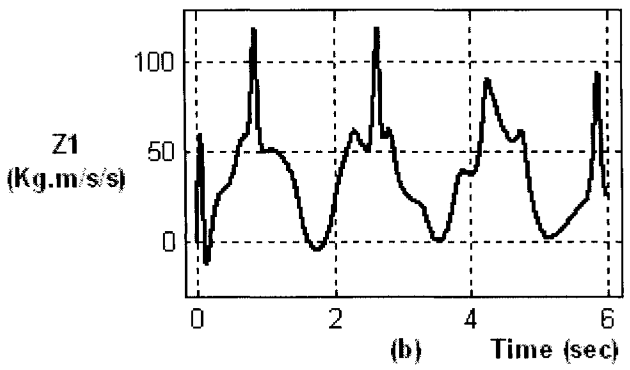
$$\mathbf{x} = [x_1 \quad y_1 \quad \dot{x}_1 \quad \dot{y}_1 \quad z_1 \quad x_2 \quad y_2 \quad \dot{x}_2 \quad \dot{y}_2 \quad z_b \quad \dot{z}_b \quad x_3 \quad y_3 \quad \dot{x}_3 \quad \dot{y}_3 \quad z_2]'$$

and starting with the initial state

$$\mathbf{x} = [-1.1 \quad -0.1 \quad 0 \quad 0 \quad 0 \quad -2.2 \quad -0.2 \quad 0 \quad 0 \quad 0 \quad 0 \quad -3.3 \quad -0.3 \quad 0 \quad 0 \quad 0]'$$

the simulation is ran for 6 seconds. The histories of the lengths w_i, states z_i and inputs v_i of the first subsystem, set of boundary algebraic equations, and the second subsystem are shown in figures 6, 7 and 8, respectively. They show that the performance of the controller is fast enough for a real-time distributed simulation.





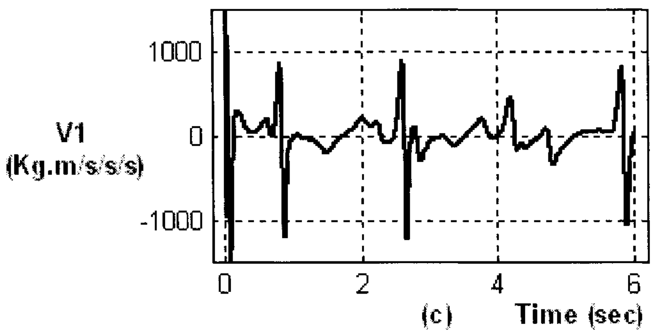
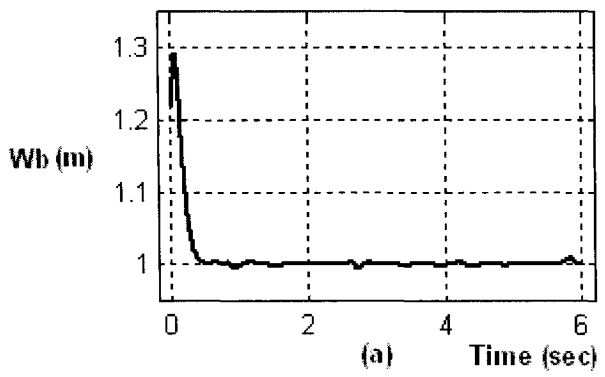
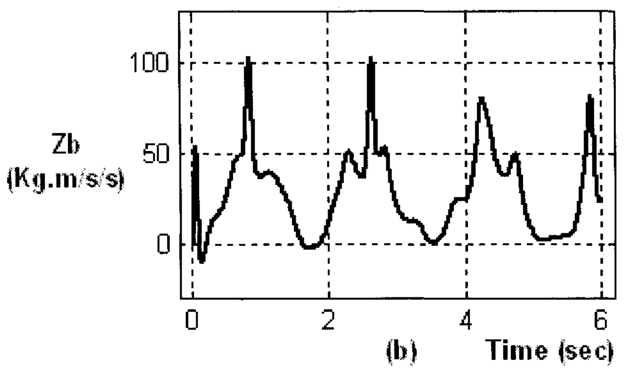


Figure 6. Decentralized control (distributed realization): (a) History of the length w_1 (corresponding to l_1), (b) History of z_1, (c) History of v_1.





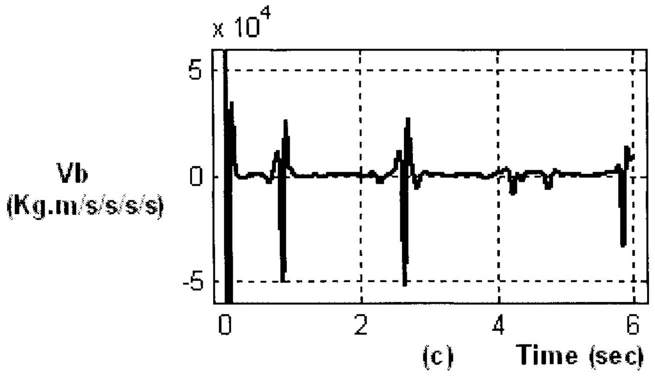


Figure 7. Decentralized control (distributed realization): (a) History of the length w_b (corresponding to l_b), (b) History of z_b, (c) History of v_b.

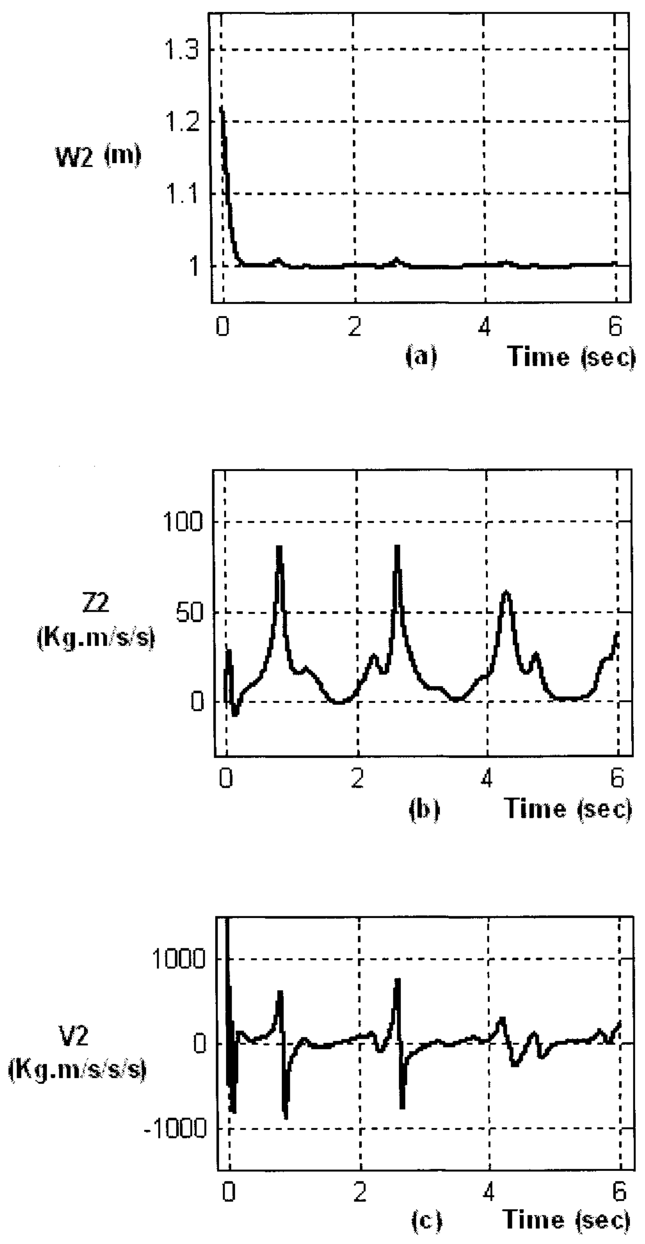


Figure 8. Decentralized control (distributed realization): (a) History of the length w_2 (corresponding to l_2), (b) History of z_2, (c) History of v_2.

It can be seen in figures 6 (a), 7 (a) and 8 (a) that the lengths w_1, w_b and w_2 converge to one perfectly after about 0.2 seconds. This convergence is reasonable compared with the following centralized control method.

Applying the centralized sliding control, the nine elements of the square matrix $\mathbf{J}_s$ are:

$$J_{s11} = \frac{-2\mu^2}{m_1}\left(x_1^2 + y_1^2\right)$$

$$J_{s12} = J_{s21} = \frac{-2\mu^2}{m_1}\left[x_1(x_1 - x_2) + y_1(y_1 - y_2)\right]$$

$$J_{s13} = J_{s31} = 0$$

$$J_{s22} = -2\mu^2 \cdot \left(\frac{1}{m_2} + \frac{1}{m_1}\right)\left[(x_2 - x_1)^2 + (y_2 - y_1)^2\right]$$

$$J_{s23} = J_{s32} = \frac{-2\mu^2}{m_2}[(x_2 - x_1)(x_2 - x_3) + (y_2 - y_1)(y_2 - y_3)]$$

$$J_{s33} = -2\mu^2 \cdot \left(\frac{1}{m_3} + \frac{1}{m_2}\right)\left[(x_3 - x_2)^2 + (y_3 - y_2)^2\right]$$

Using the same parameter values and initial state as in decentralized case, the histories of the lengths w_i, states z_i and inputs v_i of the first subsystem, set of boundary algebraic equations, and the second subsystem are shown in figures 9, 10 and 11, respectively.

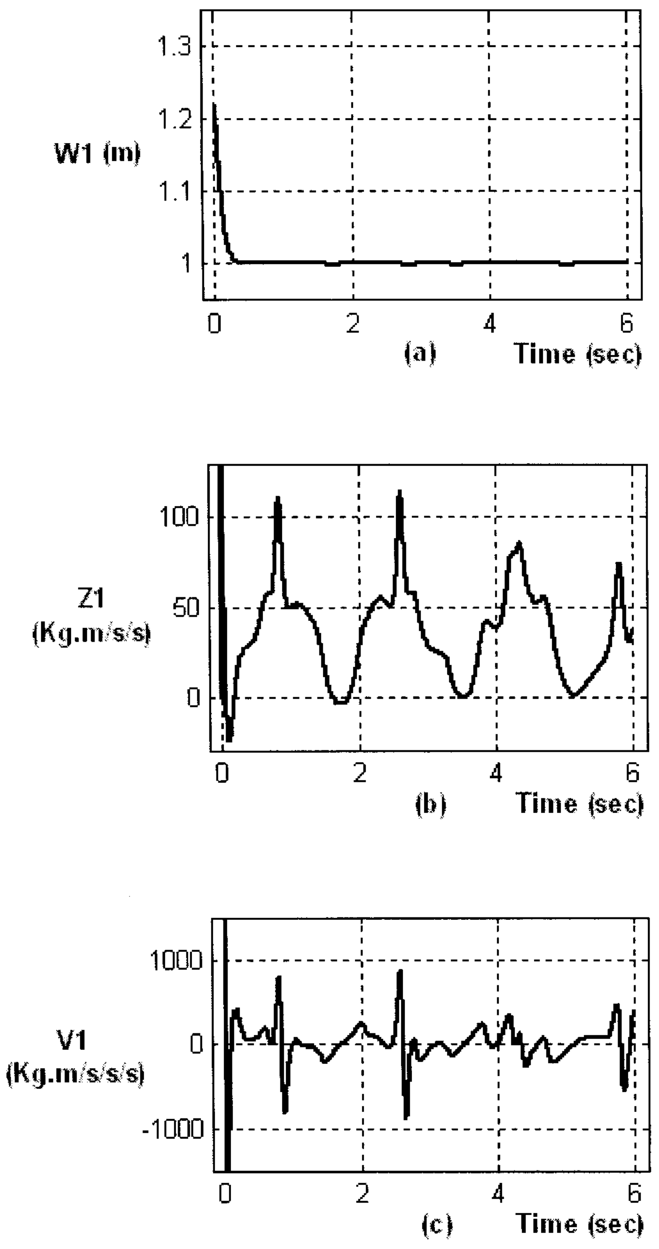


Figure 9. Centralized control: (a) History of the length w_1 (corresponding to l_1), (b) History of z_1, (c) History of v_1.

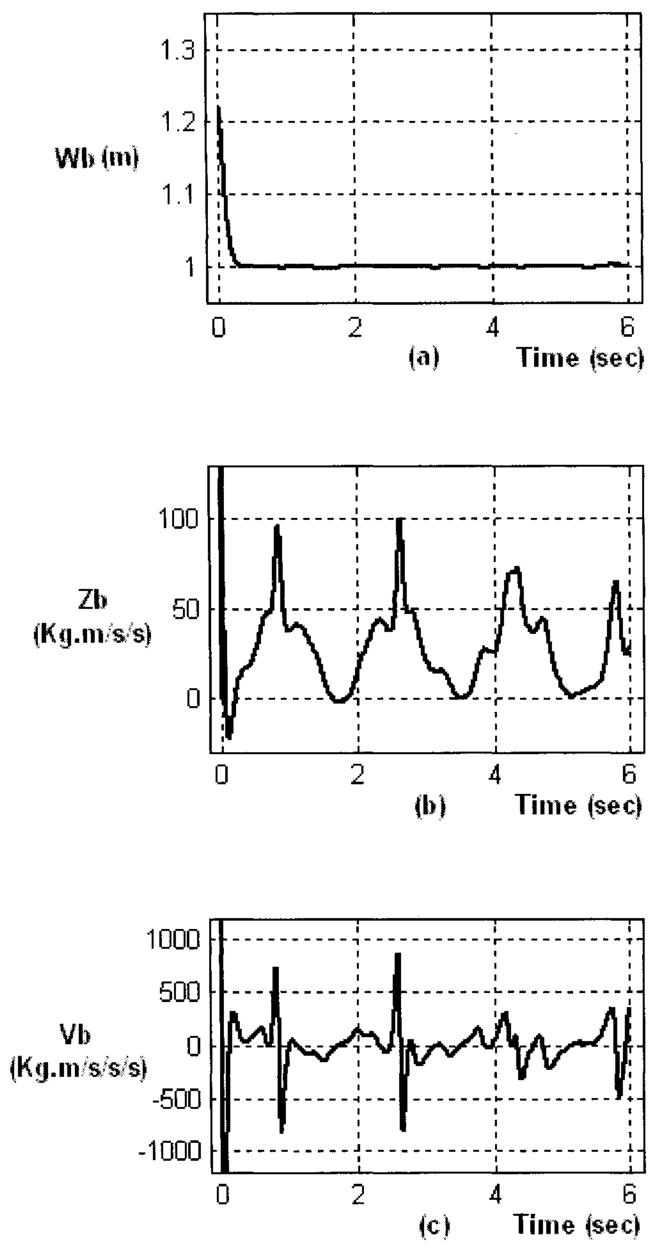


Figure 10. Centralized control: (a) History of the length w_b (corresponding to l_b), (b) History of z_b, (c) History of v_b.

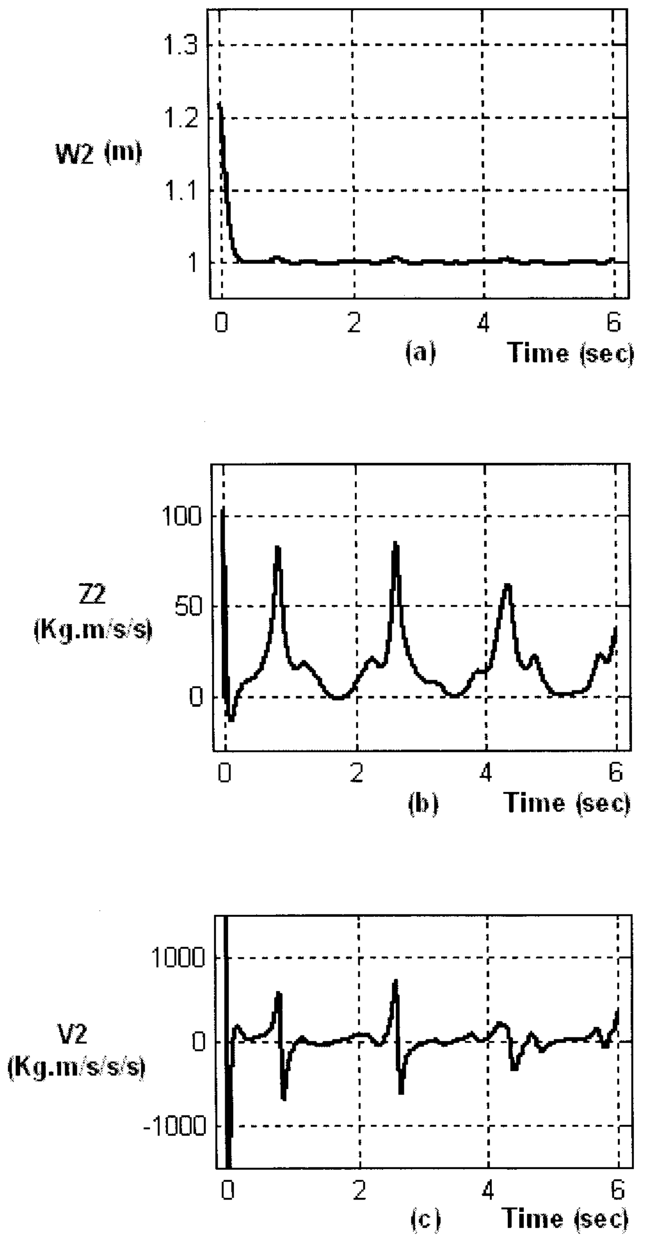


Figure 11. Centralized control: (a) History of the length w_2 (corresponding to l_2), (b) History of z_2, (c) History of v_2.

Comparing figures 6, 7 and 8 with 9, 10 and 11, respectively, shows that the performance of the decentralized sliding control in the distributed realization is quite similar to the centralized case, which confirms the validity of the analytical work. The only difference is the input v_b, which has a larger order (10 times) in the decentralized case than the centralized one.

3.2 Corollary of the Main Theorem with Adaptation Law

Using some approximation and estimation terms [36], the theorem 3 could be expressed in a different version as in the following corollary [25, 28, 31, 32, 33].

Corollary 1. Considering the stability of internal dynamics (assumption 3), in order to simulate the distributed realization {(25),(26)} in the proposition 1, using the reformulation {(34),(35)},

a) First, the decentralized sliding controllers are designed - based on the approximation of the terms $\Delta\boldsymbol{\Theta}_i$ - on the distributed computers to derive the local inputs $\mathbf{v}_i$ (i=1,...,N) for the N local nodes (34);
b) Next, the decentralized sliding controller is designed - based on adaptive Fourier series estimation of the term $\Delta\mathbf{D}_b$ - on the distributed computers to derive the boundary inputs $\mathbf{v}_b$ for the set of L boundary nodes (35);

And the overall closed-loop distributed system will be stable. From physical considerations it can be shown that all the states of the distributed model are bounded near the solution.

Proof.

Considering assumption 1, suppose that the local node (34) has a vector index with equal elements r_i, and the internal dynamics are stable, and consider the sliding manifold:

$$\mathbf{s}_i = \sum_{j=1}^{r_i}\left(\binom{r_i - 1}{j-1}\mu_i^{r_i - j}\mathbf{w}_i^{(r_i - j)}\right) \tag{74}$$

The dynamic characteristic equation of the manifold is:

$$\sum_{j=1}^{r_i}\left(\binom{r_i - 1}{j-1}\mu_i^{r_i - j}p^{(r_i - j)}\right) = 0 \tag{75}$$

and has $r_i - 1$ negative poles at $\frac{-1}{\mu_i}$ $(\mu_i > 0)$. Using equations (45) the derivatives of the manifold is:

$$\begin{aligned}\dot{\mathbf{s}}_i &= \sum_{j=2}^{r_i}\left(\binom{r_i - 1}{j-1}\mu_i^{r_i - j}\mathbf{w}_i^{(r_i - j+1)}\right) + \mu_i^{r_i - 1}\mathbf{w}_i^{(r_i)} \\ &= \sum_{j=2}^{r_i}\left(\binom{r_i - 1}{j-1}\mu_i^{r_i - j}\mathbf{w}_i^{(r_i - j+1)}\right) + \mu_i^{r_i - 1}(\boldsymbol{\alpha}_i + \boldsymbol{\beta}_i\mathbf{v}_i + \Delta\boldsymbol{\Theta}_i)\end{aligned} \tag{76}$$

According to remark 4, $\Delta\boldsymbol{\Theta}_i$ represents the effect of the interconnections of the **i-th** local node. Therefore, it is reasonable to be able to approximate this term. Assume the vector matrices $\hat{\boldsymbol{\alpha}}_i$ and $\Delta\hat{\boldsymbol{\Theta}}_i$ are the approximations of $\boldsymbol{\alpha}_i$ and $\Delta\boldsymbol{\Theta}_i$ such that $|\hat{\boldsymbol{\alpha}}_i - \boldsymbol{\alpha}_i| < \mathbf{E}_{\alpha_i}$ and $|\Delta\hat{\boldsymbol{\Theta}}_i - \Delta\boldsymbol{\Theta}_i| < \mathbf{E}_{\Delta\Theta_i}$. The control law

$$\mathbf{v}_i = \bar{\mathbf{J}}_{si}^{-1}\left(-\sum_{j=2}^{r_i}\left(\binom{r_i - 1}{j-1} \mu_i^{r_i - j}\mathbf{w}_i^{(r_i - j+1)} \right) - \mu_i^{r_i - 1}\hat{\boldsymbol{\alpha}}_i - \mu_i^{r_i - 1}\Delta\hat{\boldsymbol{\Theta}}_i \right. \\ \left. - \operatorname{diag}(\mathbf{k}_i).\mathbf{sat}(\varepsilon_i^{-1}\operatorname{diag}(\boldsymbol{\delta}_i)\mathbf{s}_i) \right) \tag{77}$$

with

$$\bar{\mathbf{J}}_{si} = \mu_i^{r_i - 1}\boldsymbol{\beta}_i \tag{78}$$

stabilizes the local node (34) if the following condition is satisfied:

$$\mathbf{k}_i \geq \mu_i^{r_i - 1}\mathbf{E}_{\alpha_i} + \mu_i^{r_i - 1}\mathbf{E}_{\Delta\Theta_i} + \boldsymbol{\gamma}_i \tag{79}$$

where all elements of the vector $\boldsymbol{\gamma}_i$ are positive.

Considering assumption 1, suppose that the boundary node (35) has a vector index with equal elements $r_b = r_i + 1$ (because of the extra integrator), and consider the sliding manifold:

$$\mathbf{s}_b = \sum_{j=1}^{r_b}\left(\binom{r_b - 1}{j-1} \mu_b^{r_b - j}\mathbf{w}_b^{(r_b - j)} \right) \tag{80}$$

The dynamic characteristic equation of the manifold is:

$$\sum_{j=1}^{r_b}\left(\binom{r_b - 1}{j-1} \mu_b^{r_b - j} p^{(r_b - j)} \right) = 0 \tag{81}$$

and has $r_b - 1$ negative poles at $\frac{-1}{\mu_b}$ $(\mu_b > 0)$. Using equations (45), the derivatives of the manifold is

$$\begin{aligned}\dot{\mathbf{s}}_b &= \sum_{j=2}^{r_b}\left(\binom{r_b-1}{j-1}\mu_b^{r_b-j}\mathbf{w}_b^{(r_b-j+1)}\right) + \mu_b^{r_b-1}\mathbf{w}_b^{(r_b)} \\ &= \sum_{j=2}^{r_b}\left(\binom{r_b-1}{j-1}\mu_b^{r_b-j}\mathbf{w}_b^{(r_b-j+1)}\right) + \mu_b^{r_b-1}\left(\boldsymbol{\alpha}_b + \boldsymbol{\beta}_b\mathbf{v}_b + \Delta\mathbf{D}_b\right)\end{aligned} \tag{82}$$

According to remark 5, $\Delta\mathbf{D}_b$ represents the effects of the local inputs, as well as their derivatives, on the set of boundary nodes. Therefore, it may be impossible and costly to find an approximation term for it, according to the decentralized nature of the controller. Instead, it is reasonable and possible to estimate this term by using the Fourier series (52) in remark 7. Assume the vector matrix $\hat{\boldsymbol{\alpha}}_b$ is the estimation of $\boldsymbol{\alpha}_b$ such that $|\hat{\boldsymbol{\alpha}}_b - \boldsymbol{\alpha}_b| < \mathbf{E}_{\boldsymbol{\alpha}_b}$, and replace the equality (52) for the term $\Delta\mathbf{D}_b$. The control law

$$\begin{aligned}\mathbf{v}_b = \bar{\mathbf{J}}_{sb}^{-1}\Bigg(&-\sum_{j=2}^{r_b}\left(\binom{r_b-1}{j-1}\mu_b^{r_b-j}\mathbf{w}_b^{(r_b-j+1)}\right) - \mu_b^{r_b-1}\hat{\boldsymbol{\alpha}}_b - \mu_b^{r_b-1}\hat{\boldsymbol{\Psi}}_b^T\cdot\mathbf{F}(t) \\ &- \mathrm{diag}(\mathbf{k}_b).\mathbf{sat}(\varepsilon_b^{-1}\mathrm{diag}(\boldsymbol{\delta}_b)\mathbf{s}_b)\Bigg)\end{aligned} \tag{83}$$

where $\hat{\boldsymbol{\Psi}}_b$ is the estimation of the real $\boldsymbol{\Psi}_b$, with

$$\bar{\mathbf{J}}_{sb} = \mu^{r_b-1}\boldsymbol{\beta}_b(.) \tag{84}$$

and the adaptation law

$$\dot{\hat{\boldsymbol{\Psi}}}_b = \frac{d}{dt}\left(\hat{\boldsymbol{\Psi}}_b - \boldsymbol{\Psi}_b\right) = \dot{\tilde{\boldsymbol{\Psi}}}_b = \mu^{r_b-1}\boldsymbol{\Omega}_1\mathbf{F}(t)\mathbf{s}_b^T\boldsymbol{\Omega}_2 \tag{85}$$

where $\mathbf{\Omega}_1$ and $\mathbf{\Omega}_2$ are the diagonal positive definite adaptation matrices, stabilizes the set of boundary nodes (35) if the following condition is satisfied:

$$\mathbf{k}_b \geq \mu_b^{r_b-1}\mathbf{E}_{\alpha_b} + \gamma_b \tag{86}$$

where all elements of the vector γ_b are positive.

The control laws (77) and (83) designed so far will stabilize the distributed realization $\{(34),(35)\}$. Consider the quadratic Lyapunov function:

$$V_t = \frac{1}{2}\sum_{i=1}^{N}\mathbf{s}_i^T\mathbf{s}_i + \frac{1}{2}\mathbf{s}_b^T\mathbf{s}_b + \frac{1}{2}\mathrm{trace}\left(\mathbf{\Omega}_2^{-1}\tilde{\mathbf{\Psi}}_b^T\mathbf{\Omega}_1^{-1}\tilde{\mathbf{\Psi}}_b\right) \geq 0 \tag{87}$$

The last terms represent the adaptive coefficients. Using the manifold derivatives (76) and (82), and applying the control inputs (77) and (83), the derivative of the Lyapunov function is:

$$\begin{aligned}\frac{dV_t}{dt} &= \sum_{i=1}^{N}\mathbf{s}_i^T\dot{\mathbf{s}}_i + \mathbf{s}_b^T\dot{\mathbf{s}}_b + \mathrm{trace}\left(\mathbf{\Omega}_2^{-1}\tilde{\mathbf{\Psi}}_b^T\mathbf{\Omega}_1^{-1}\dot{\tilde{\mathbf{\Psi}}}_b\right)\\ &= \sum_{i=1}^{N}\mathbf{s}_i^T\left(\mu_i^{r_i-1}\left(\boldsymbol{\alpha}_i(.) - \hat{\boldsymbol{\alpha}}_i(.)\right) + \mu_i^{r_i-1}\left(\Delta\mathbf{\Theta}_i - \Delta\hat{\mathbf{\Theta}}_i\right) - \mathrm{diag}(\mathbf{k}_i)\mathbf{sat}(\varepsilon_i^{-1}\mathrm{diag}(\boldsymbol{\delta}_i)\mathbf{s}_i)\right)\\ &+ \mathbf{s}_b^T\left(\mu_b^{r_b-1}\left(\boldsymbol{\alpha}_b(.) - \hat{\boldsymbol{\alpha}}_b(.)\right) + \mu_b^{r_b-1}\left(\left(\mathbf{\Psi}_b^T - \hat{\mathbf{\Psi}}_b^T\right)\cdot\mathbf{F}(t)\right) - \mathrm{diag}(\mathbf{k}_b)\mathbf{sat}(\varepsilon_b^{-1}\mathrm{diag}(\boldsymbol{\delta}_b)\mathbf{s}_b)\right)\\ &+ \mathrm{trace}\left(\mathbf{\Omega}_2^{-1}\tilde{\mathbf{\Psi}}_b^T\mathbf{\Omega}_1^{-1}\dot{\tilde{\mathbf{\Psi}}}_b\right)\end{aligned} \tag{88}$$

For the last term of the derivative above, the adaptation law (85) is used to simplify as follow:

$$\begin{aligned}\operatorname{trace}\left(\boldsymbol{\Omega}_2^{-1}\widetilde{\boldsymbol{\Psi}}_b^T\boldsymbol{\Omega}_1^{-1}\dot{\widetilde{\boldsymbol{\Psi}}}_b\right) &= \operatorname{trace}\left(\boldsymbol{\Omega}_2^{-1}\widetilde{\boldsymbol{\Psi}}_b^T\boldsymbol{\Omega}_1^{-1}\left(\mu^{r_b-1}\boldsymbol{\Omega}_1\mathbf{F}(t)\mathbf{s}_b^T\boldsymbol{\Omega}_2\right)\right)\\ &= \mu^{r_b-1}\operatorname{trace}\left(\boldsymbol{\Omega}_2^{-1}\widetilde{\boldsymbol{\Psi}}_b^T\mathbf{F}(t)\mathbf{s}_b^T\boldsymbol{\Omega}_2\right)\\ &= \mu^{r_b-1}\operatorname{trace}\left(\mathbf{s}_b^T\boldsymbol{\Omega}_2\boldsymbol{\Omega}_2^{-1}\widetilde{\boldsymbol{\Psi}}_b^T\mathbf{F}(t)\right)\\ &= \mu^{r_b-1}\operatorname{trace}\left(\mathbf{s}_b^T\widetilde{\boldsymbol{\Psi}}_b^T\mathbf{F}(t)\right)\\ &= \mu^{r_b-1}\mathbf{s}_b^T\widetilde{\boldsymbol{\Psi}}_b^T\mathbf{F}(t)\end{aligned} \tag{89}$$

Since $\widetilde{\boldsymbol{\Psi}}_b = \hat{\boldsymbol{\Psi}}_b - \boldsymbol{\Psi}_b$, using the simplification (89) in the derivative (88) results in:

$$\begin{aligned}\frac{dV_t}{dt} &= \sum_{i=1}^{N}\mathbf{s}_i^T\left(\mu_i^{r_i-1}\left(\boldsymbol{\alpha}_i(.)-\hat{\boldsymbol{\alpha}}_i(.)\right)+\mu_i^{r_i-1}\left(\Delta\boldsymbol{\Theta}_i-\Delta\hat{\boldsymbol{\Theta}}_i\right)-\operatorname{diag}(\mathbf{k}_i)\mathbf{sat}(\varepsilon_i^{-1}\operatorname{diag}(\boldsymbol{\delta}_i)\mathbf{s}_i)\right)\\ &+\mathbf{s}_b^T\left(\mu_b^{r_b-1}\left(\boldsymbol{\alpha}_b(.)-\hat{\boldsymbol{\alpha}}_b(.)\right)+\mu_b^{r_b-1}\left(-\widetilde{\boldsymbol{\Psi}}_b^T\mathbf{F}(t)\right)-\operatorname{diag}(\mathbf{k}_b)\mathbf{sat}(\varepsilon_b^{-1}\operatorname{diag}(\boldsymbol{\delta}_b)\mathbf{s}_b)\right)\\ &+\mu^{r_b-1}\mathbf{s}_b^T\widetilde{\boldsymbol{\Psi}}_b^T\mathbf{F}(t)\end{aligned} \tag{90}$$

Now, canceling out the two terms related to the Fourier expansion and adaptive coefficient errors $\widetilde{\boldsymbol{\Psi}}_b$ results in:

$$\begin{aligned}\frac{dV_t}{dt} &= \sum_{i=1}^{N}\mathbf{s}_i^T\left(\mu_i^{r_i-1}\left(\boldsymbol{\alpha}_i(.)-\hat{\boldsymbol{\alpha}}_i(.)\right)+\mu_i^{r_i-1}\left(\Delta\boldsymbol{\Theta}_i-\Delta\hat{\boldsymbol{\Theta}}_i\right)-\operatorname{diag}(\mathbf{k}_i)\mathbf{sat}(\varepsilon_i^{-1}\operatorname{diag}(\boldsymbol{\delta}_i)\mathbf{s}_i)\right)\\ &+\mathbf{s}_b^T\left(\mu_b^{r_b-1}\left(\boldsymbol{\alpha}_b(.)-\hat{\boldsymbol{\alpha}}_b(.)\right)-\operatorname{diag}(\mathbf{k}_b)\mathbf{sat}(\varepsilon_b^{-1}\operatorname{diag}(\boldsymbol{\delta}_b)\mathbf{s}_b)\right)\end{aligned} \tag{91}$$

Therefore the upper bounds $|\hat{\boldsymbol{\alpha}}_i-\boldsymbol{\alpha}_i|<\mathbf{E}_{\alpha_i}$, $|\Delta\hat{\boldsymbol{\Theta}}_i-\Delta\boldsymbol{\Theta}_i|<\mathbf{E}_{\Delta\Theta_i}$ and $|\hat{\boldsymbol{\alpha}}_b-\boldsymbol{\alpha}_b|<\mathbf{E}_{\Delta\Theta_i}$ certify that:

$$\begin{aligned}\frac{dV_t}{dt} &\le \sum_{i=1}^{N}\mathbf{s}_i^T\left(\mu_i^{r_i-1}\mathbf{E}_{\alpha_i}+\mu_i^{r_i-1}\mathbf{E}_{\Delta\Theta_i}-\operatorname{diag}(\mathbf{k}_i)\mathbf{sat}(\varepsilon_i^{-1}\operatorname{diag}(\boldsymbol{\delta}_i)\mathbf{s}_i)\right)\\ &+\mathbf{s}_b^T\left(\mu_b^{r_b-1}\mathbf{E}_{\Delta\Theta_b}-\operatorname{diag}(\mathbf{k}_b)\mathbf{sat}(\varepsilon_b^{-1}\operatorname{diag}(\boldsymbol{\delta}_b)\mathbf{s}_b)\right)\end{aligned} \tag{92}$$

Now, considering the conditions (79) and (86) results in:

$$\frac{dV_t}{dt} \le \sum_{i=1}^{N} \mathbf{s}_i^T\left(-\mathrm{diag}(\boldsymbol{\gamma}_i)\mathbf{sat}(\varepsilon_i^{-1}\mathrm{diag}(\boldsymbol{\delta}_i)\mathbf{s}_i)\right) + \mathbf{s}_b^T\left(-\mathrm{diag}(\boldsymbol{\gamma}_b)\mathbf{sat}(\varepsilon_b^{-1}\mathrm{diag}(\boldsymbol{\delta}_b)\mathbf{s}_b)\right) \tag{93}$$

Since all elements of the vectors $\boldsymbol{\gamma}_i$ and $\boldsymbol{\gamma}_b$ are positive, it is concluded that

$$\frac{dV_t}{dt} \le 0 \tag{94}$$

So the stability of the closed-loop distributed realization is guaranteed.

□

Remark 8. Consider the adaptation law (85) as:

$$\dot{\hat{\boldsymbol{\Psi}}}_b = \mu^{r_b-1}\boldsymbol{\Omega}_1\mathbf{F}(t)\mathbf{s}_b^T\boldsymbol{\Omega}_2 \tag{95}$$

The special configuration of the adaptation law including $\boldsymbol{\Omega}_1$ and $\boldsymbol{\Omega}_2$ is for the flexibility to choose the arbitrary adaptation coefficient for each term individually. For the special case in which $\boldsymbol{\Psi}_b = [\psi_{b1} \quad \psi_{b2}]$, $\mathbf{F}(t) = [f_1(t) \quad f_2(t) \quad f_3(t)]^T$ and $\mathbf{s}_b = [s_{b1} \quad s_{b2}]^T$, the adaptation law is:

$$\begin{bmatrix} \dot{\hat{\psi}}_{b11} & \dot{\hat{\psi}}_{b21} \\ \dot{\hat{\psi}}_{b12} & \dot{\hat{\psi}}_{b22} \\ \dot{\hat{\psi}}_{b13} & \dot{\hat{\psi}}_{b23} \end{bmatrix} = \mu^{r_b-1}\begin{bmatrix} \omega_{11} & 0 & 0 \\ 0 & \omega_{12} & 0 \\ 0 & 0 & \omega_{13} \end{bmatrix} \cdot \begin{bmatrix} f_1(t) \\ f_2(t) \\ f_3(t) \end{bmatrix} \cdot [s_{b1} \quad s_{b2}] \cdot \begin{bmatrix} \omega_{21} & 0 \\ 0 & \omega_{22} \end{bmatrix} \tag{96}$$

It results in

$$\begin{bmatrix} \dot{\hat{\psi}}_{b11} & \dot{\hat{\psi}}_{b21} \\ \dot{\hat{\psi}}_{b12} & \dot{\hat{\psi}}_{b22} \\ \dot{\hat{\psi}}_{b13} & \dot{\hat{\psi}}_{b23} \end{bmatrix} = \begin{bmatrix} \omega_{11}\omega_{21}\mu^{r_b-1}f_1(t)s_{b1} & \omega_{11}\omega_{22}\mu^{r_b-1}f_1(t)s_{b2} \\ \omega_{12}\omega_{21}\mu^{r_b-1}f_2(t)s_{b1} & \omega_{12}\omega_{22}\mu^{r_b-1}f_2(t)s_{b2} \\ \omega_{13}\omega_{21}\mu^{r_b-1}f_3(t)s_{b1} & \omega_{13}\omega_{22}\mu^{r_b-1}f_3(t)s_{b2} \end{bmatrix} \tag{97}$$

It shows that the combinations of $\omega_{1i}\omega_{2j}$ (i=1,2,3 , j=1,2) creates 6 arbitrary different coefficients for the 6 updating adaptive laws $\dot{\hat{\psi}}_{bji}$. So there is no restriction on choosing the different coefficients for different parameters. Moreover, the corresponding term in the Lyapunov function (87) is:

$$\frac{1}{2}\operatorname{trace}\left(\mathbf{\Omega}_2^{-1}\tilde{\mathbf{\Psi}}_b^T\mathbf{\Omega}_1^{-1}\tilde{\mathbf{\Psi}}_b\right) \tag{98}$$

For the special case above, this term is:

$$\frac{1}{2}\operatorname{trace}\left(\begin{bmatrix} \omega_{21} & 0 \\ 0 & \omega_{22} \end{bmatrix}^{-1} \cdot \begin{bmatrix} \tilde{\psi}_{b11} & \tilde{\psi}_{b12} & \tilde{\psi}_{b13} \\ \tilde{\psi}_{b21} & \tilde{\psi}_{b22} & \tilde{\psi}_{b23} \end{bmatrix} \cdot \begin{bmatrix} \omega_{11} & 0 & 0 \\ 0 & \omega_{12} & 0 \\ 0 & 0 & \omega_{13} \end{bmatrix}^{-1} \cdot \begin{bmatrix} \tilde{\psi}_{b11} & \tilde{\psi}_{b21} \\ \tilde{\psi}_{b12} & \tilde{\psi}_{b22} \\ \tilde{\psi}_{b13} & \tilde{\psi}_{b23} \end{bmatrix}\right) \tag{99}$$
$$= \frac{1}{2}\sum_{i=1}^{3}\sum_{j=1}^{2}\omega_{1i}\omega_{2j}\tilde{\psi}_{bji}^2$$

Which shows that this term is a linear quadratic function of all the estimation errors $\tilde{\psi}_{bji}$ (i=1,2,3 , j=1,2).

Remark 9. In most applications, the matrix $\bar{\mathbf{J}}_{sb}$ (related to the set of boundary nodes) is block-diagonal, and so the inverse of this matrix can be calculated in a distributed fashion, as well. The following example is more illustrative.

Example 4. Consider figure 12 which is the application to animation of a deformable surface.

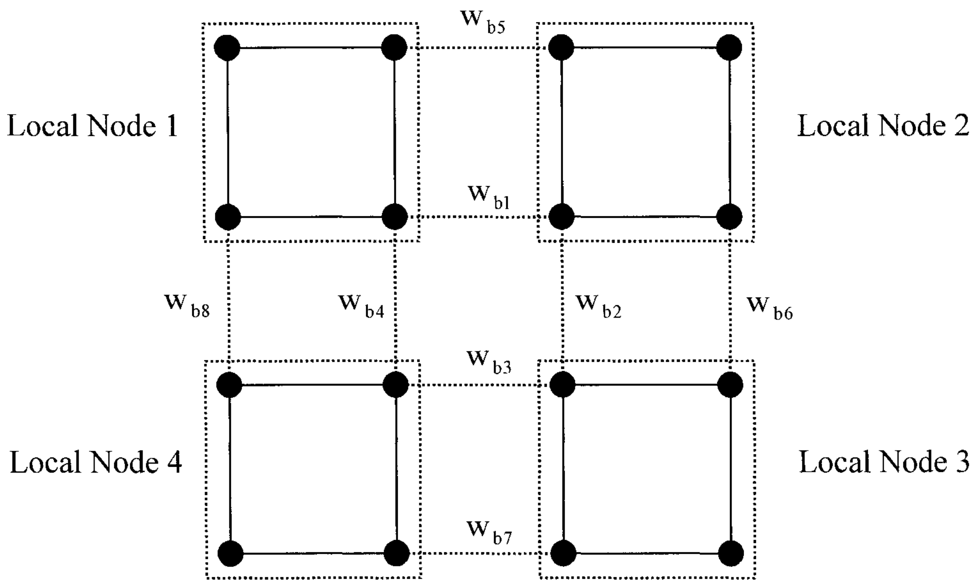


Figure 12. Distributed model of a deformable surface: The deformable surface, which is approximated by 16 particles, is divided into 4 local nodes with 4 particles each.

In this figure, a deformable surface approximated by 16 particles is divided into 4 local nodes with 4 particles each. The set of boundary nodes includes $\{(v_{bi}, w_{bi}) \mid i = 1,2,\ldots,8\}$.

The matrix $\bar{\mathbf{J}}_{sb}$ for this example is:

$$\bar{\mathbf{J}}_{sb} = \begin{bmatrix} \bar{J}_{sb11} & \bar{J}_{sb12} & 0 & \bar{J}_{sb14} & 0 & 0 & 0 & 0 \\ \bar{J}_{sb21} & \bar{J}_{sb22} & \bar{J}_{sb23} & 0 & 0 & 0 & 0 & 0 \\ 0 & \bar{J}_{sb32} & \bar{J}_{sb33} & \bar{J}_{sb34} & 0 & 0 & 0 & 0 \\ \bar{J}_{sb41} & 0 & \bar{J}_{sb43} & \bar{J}_{sb44} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \bar{J}_{sb55} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \bar{J}_{sb66} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \bar{J}_{sb77} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \bar{J}_{sb88} \end{bmatrix} \tag{100}$$

It shows that the matrix $\bar{\mathbf{J}}_{sb}$ is block-diagonal. Since it is divided into several blocks, the inverse of each block can be calculated on a separate computer. Hence, the block-diagonal structure of $\bar{\mathbf{J}}_{sb}$ is helpful for saving the inverse calculation time, specifically in cases that we are dealing with a large set of boundary nodes which is the result of the large-scale structure of the DAE system. For the application to animation of a deformable surface, the maximum dimensions of a block on the block-diagonal structure of $\bar{\mathbf{J}}_{sb}$ is 4×4. This 4×4 block appears on the diagonal of the matrix in (100). It pertains to the boundary algebraic equations $\{w_{b1}, w_{b2}, w_{b3}, w_{b4}\}$ which are connecting the four neighboring local nodes on their approaching corners (as in figure 12).

The corresponding matrices $\bar{\mathbf{J}}_{si}$ and $\bar{\mathbf{J}}_{sb}$ in (78) and (84) for each local node (34) and the boundary node (35), are calculated separately. Calculating the inverse transform of $\bar{\mathbf{J}}_{si}$ and $\bar{\mathbf{J}}_{sb}$ on separate computers (distributed method) is considerably less time consuming than the inverse of $\mathbf{J}_s$ matrix of the overall system on a single computer.

3.2.1 Advantage of the Adaptive Law

Remark 10. The only difference between theorem 3 and corollary 1 is the approximation of the terms $\Delta\mathbf{\Theta}_i$ and the adaptive Fourier series estimation of the term $\Delta\mathbf{D}_b$. These extra terms help to decrease the sliding gains $\mathbf{k}_i$ and $\mathbf{k}_b$ considerably, since the following inequalities hold intuitively:

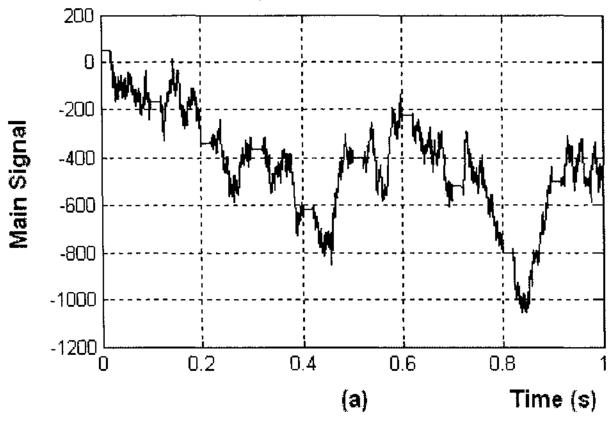
$$\max\left(|\Delta\hat{\mathbf{\Theta}}_i - \Delta\mathbf{\Theta}_i|\right) << \max\left(|\Delta\mathbf{\Theta}_i|\right) \quad \Rightarrow \quad \mathbf{E}_{\Delta\Theta_i} << \mathbf{B}_{\Delta\Theta_i} \tag{101}$$

and

$$\max\left(|\hat{\mathbf{\Psi}}_b^T \mathbf{F}(t) - \Delta\mathbf{D}_b|\right) << \max\left(|\Delta\mathbf{D}_b|\right) \Rightarrow \quad \mathbf{E}_{\Delta D_b} << \mathbf{B}_{\Delta D_b} \tag{102}$$

As a justification, the following example illustrates how inequality (102) holds.

Example 5. In this simulation, the random signal in figure 13 (a) is produced. The amplitude of the signal is of order 3 ($B_{\Delta D_b} = 10^3$). This signal is approximated by the Fourier series expansion up to the first 5 parameters a_0, a_1, b_1, a_2 and b_2. Figure 13 (b) shows the error between the main signal in 13 (a) and the adaptive Fourier series estimation of the main signal. The amplitude of the error signal is of order 2 ($E_{\Delta D_b} = 10^2$). It shows that using only 5 adaptive parameters, the amplitude is decreased by 10 times ($E_{\Delta D_b} = B_{\Delta D_b} \div 10$). This important result is useful to decrease the order of the sliding gains $\mathbf{k}_i$ and $\mathbf{k}_b$ accordingly.



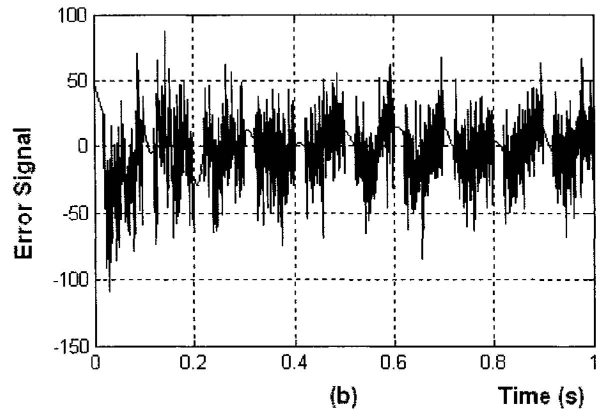


Figure 13. (a) A random signal, (b) Error of the estimation of the main signal with adaptive Fourier series expansion with 5 parameters.

3.3 Parameter Selection for Designing the Controller

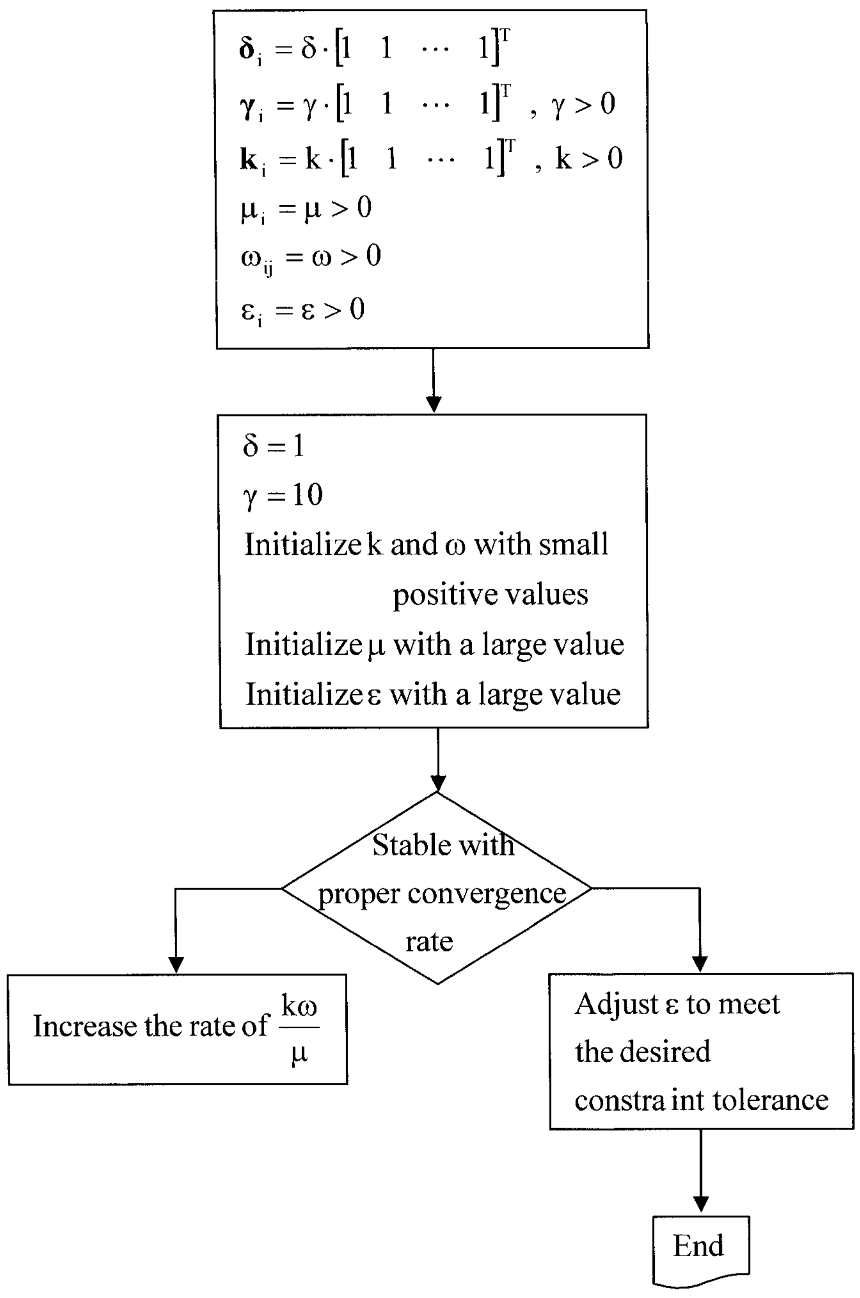
In this section, the strategy for selecting the controller parameters is mentioned. All the designing parameters are listed below. The index i=1,2,...,N where N is number of local nodes, and the index b refers to the boundary node.

ε_i	Boundary layer thickness of the i-th local node
ε_b	Boundary layer thickness of the boundary node
$-\frac{1}{\mu_i}$, $\mu_i > 0$	Characteristic equation poles of the sliding surface of the i-th local node
$-\frac{1}{\mu_b}$, $\mu_b > 0$	Characteristic equation poles of the sliding surface of the boundary node
$\boldsymbol{\delta}_i$	Relative importance vector of the i-th local node
$\boldsymbol{\delta}_b$	Relative importance vector of the boundary node
$\boldsymbol{\kappa}_i$	Gain vector of the i-th local node
$\boldsymbol{\kappa}_b$	Gain vector of the boundary node
$\boldsymbol{\gamma}_i$	Stability margin of the i-th local node
$\boldsymbol{\gamma}_b$	Stability margin of the boundary node

Because of the structural similarity between all the nodes (local and boundary nodes), and to avoid the complexity involved with parameter selection, all the corresponding parameters are chosen equal regardless of which node they belong to.

$$\begin{aligned}
&\boldsymbol{\delta}_i = \delta \cdot [1 \quad 1 \quad \cdots \quad 1] \\
&\boldsymbol{\gamma}_i = \gamma \cdot [1 \quad 1 \quad \cdots \quad 1] \;, \; \gamma > 0 \\
&\mathbf{k}_i = k \cdot [1 \quad 1 \quad \cdots \quad 1] \;, \; k > 0 \\
&\mu_i = \mu > 0 \\
&\omega_{ij} = \omega > 0 \\
&\varepsilon_i = \varepsilon > 0
\end{aligned} \tag{103}$$

The flow chart in figure 14 is the experimental strategy for selecting the corresponding parameters.



$\boldsymbol{\delta}_i = \delta \cdot [1\ 1\ \cdots\ 1]^T$
$\boldsymbol{\gamma}_i = \gamma \cdot [1\ 1\ \cdots\ 1]^T$, $\gamma > 0$
$\mathbf{k}_i = k \cdot [1\ 1\ \cdots\ 1]^T$, $k > 0$
$\mu_i = \mu > 0$
$\omega_{ij} = \omega > 0$
$\varepsilon_i = \varepsilon > 0$
$\delta = 1$
$\gamma = 10$
Initialize k and ω with small positive values
Initialize μ with a large value
Initialize ε with a large value
Stable with proper convergence rate
Increase the rate of $\frac{k\omega}{\mu}$
Adjust ε to meet the desired constra int tolerance
End

Figure 14. The flow chart for selecting the controller parameters.

The ratio $\frac{k\omega}{\mu}$ is a fair measurement of the convergence. Initially, the algorithm starts with small values for k and ω, and large value for μ, so that the ration $\frac{k\omega}{\mu}$ is small enough to start the algorithm. Then, these values are adjusted experimentally so that this ratio increases gradually. This could be done by playing with the parameter values within a reasonable rage. Once the stability is guaranteed, it is time to find the proper value for ε, which was initially set to a small value (to represents a fair approximation for the sign function). It is important that after the time constant μ in the sliding manifolds is chosen, the parameter ε can be adjusted to meet the desired constraint tolerances.

In this chapter, decentralized controllers were designed for the local and boundary nodes in order to stabilize the overall DAE system. The advantage of adaptive Fourier series approximation was explained, and a flow chart was presented in order to select the parameters of the decentralized sliding mode controller. In the next chapter the new approach is applied for realization of deformable surface problems.

4 . Application to Animation of a Deformable Surface

In this section, the designing method developed so far is applied to the animation of a deformable surface. Since the Jacobian matrix has a high dimension, corresponding to the dimensions of the deformable surface, this is a suitable example to illustrate the effectiveness of the controllers designed.

4.1 Modeling of a Deformable Surface

In simulation, a deformable surface is modeled as a collection of distributed masses connected together by rigid or flexible connections (figure 15).

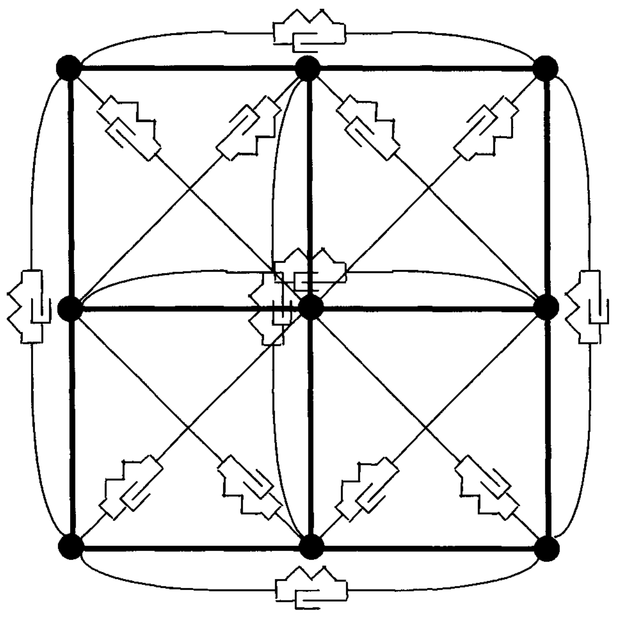


Figure 15. The spring-damper structure of a particle-based model for simulation of a deformable surface.

The dynamic equation of each particle is:

$$\ddot{\mathbf{x}}_i = \frac{1}{m_i}\left(\sum_j z_j \hat{\mathbf{r}}_{ij} + \vec{\mathbf{F}}_{int,i} + \vec{\mathbf{F}}_{ext,i}\right) \quad , \quad i = 1,\dots,n_p$$
$$\dot{z}_j = v_j \tag{104}$$

where n_p is the number of particles, z_j is the rigid force between the i-th particle and the j-th linked particle in the neighborhood (of the i-th particle), and $\hat{\mathbf{r}}_{ij}$ is the unit vector from particle i to particle j, which is at the other end of the j-th rigid link. $\vec{\mathbf{F}}_{int}$ is the summation of all the internal flexible force vectors, which corresponds to the spring-damper structure in figure 15. $\vec{\mathbf{F}}_{ext}$ is the summation of all the external forces, e.g. gravity and wind force. The constraints are:

$$w_k = L_k - L_{k0} \quad , \quad k = 1,\dots,n_l \tag{105}$$

where n_l is the number of links, and L_k and L_{k0} are the real length and desired length of the k-th link, respectively. Therefore, the deformable surface is structurally approximate by a $n_s \times n_s$ square of particles (i.e. n_s particles on each side of the square). The number of links (algebraic equations) is equal to $2n_s(n_s - 1)$. In case of centralized simulation, the Jacobian $\mathbf{J}_s$ is a $[2n_s(n_s - 1)] \times [2n_s(n_s - 1)]$ matrix, and computing the inverse of this matrix gets more time consuming as n_s increases. In practice when n_s exceeds 8, the $\mathbf{J}_s$ inverse computation timing becomes too long to meet the timing constraints of a real time simulation. So the distributed simulation technique proposed so far will take care of the timing constraints in a real time simulation.

The interpretation of the distributed model is to cut the deformable surface along the set of boundary algebraic equations, which consists up of the algebraic equations connecting the local nodes of the deformable surface (figure 16). Because of the structural similarity between the dynamic equations of different segments of the deformable surface, it is arbitrary and flexible to choose the set of boundary algebraic equations. Meanwhile the main factor to determine the choice of boundary algebraic equation is that the number of particles in each node (segment) is confined to a maximum of 8×8 square, similar to the restriction in the centralized case. It means that the restriction on the dimension of the Jacobian matrix $\bar{\mathbf{J}}_{sk}$ of the k-th node in the distributed simulation is inherited from the restriction on the dimension of the Jacobian matrix $\mathbf{J}_s$ in the centralized simulation. To address this restriction, the boundary algebraic equations are chosen in a way to divide the whole square structure of the deformable surface into 8×8 sub-squares, which are consistent to form the whole square structure, do not have overlap with each other, and are completely connected by the set of boundary algebraic equations. This explicit kind of division is shown in figure 12. Each local node of the deformable surface is simulated on a separate computer, while the calculations involved with the boundary node are done on a separate computer as well.

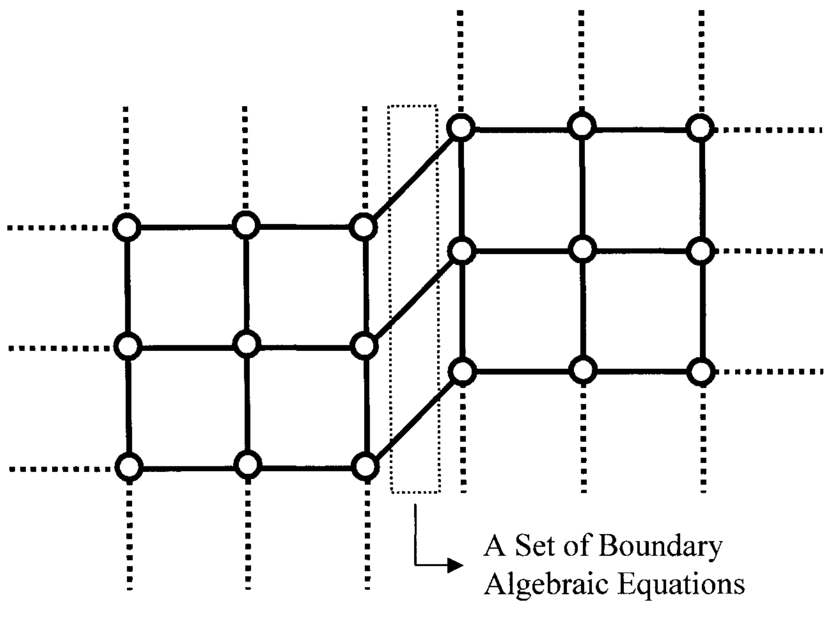


Figure 16. The physical interpretation of the distributed model of a deformable surface.

For the k-th node, the Jacobian matrix is calculated by:

$$
\left[\bar{\mathbf{J}}_{sk}\right]_{i,j} = \begin{cases}
-2\mu^{r_k-1}\vec{\mathbf{r}}_{i12}\cdot\vec{\mathbf{r}}_{i12}\left(\dfrac{1}{m_{i1}}+\dfrac{1}{m_{i2}}\right) & \text{if } i=j \text{ and none of the two end particles} \\
 & \text{has an acceleration constraint} \\
-2\mu^{r_k-1}\dfrac{\vec{\mathbf{r}}_{i12}\cdot\vec{\mathbf{r}}_{i12}}{m_{i,nc}} & \text{if } i=j \text{ and one of the two end particles} \\
 & \text{has an acceleration constraint} \\
-2\mu^{r_k-1}\dfrac{\vec{\mathbf{r}}_{ci}\cdot\vec{\mathbf{r}}_{cj}}{m_{(i,j)}} & \text{if } i\neq j \text{ and links i and j have one end} \\
 & \text{particle in common} \\
0 & \text{if } i\neq j \text{ and links i and j have noe end} \\
 & \text{particle in common}
\end{cases} \tag{106}
$$

where m_{i1} and m_{i2} are the two end particles of the link i, $m_{i,nc}$ is the end particle of the link i which does not have an acceleration constraint, and $m_{(i,j)}$ is the common end particle of the links i and j. $\vec{\mathbf{r}}_{i12}$ is the vector from the end 1 to 2 of link i, and $\vec{\mathbf{r}}_{ci}$ and $\vec{\mathbf{r}}_{ci}$ are the vectors from the common end to the other ends of the links i and j, respectively. The acceleration constraint pertains to the case that one particle is tightly fixed not to make any movement.

For the local nodes, since the number of particles in each node is confined to a maximum of $n_s \times n_s = 8 \times 8$ square, the corresponding Jacobian matrix $\overline{\mathbf{J}}_{sk}$ is a 112×112 square matrix ($2n_s(n_s - 1) = 2 \times 8 \times 7 = 112$). This is a reasonable dimension for a matrix to be inversed on a computer while the requirements of a real time simulation are met. For the boundary node, the dimension of the Jacobian matrix $\overline{\mathbf{J}}_{sb}$ is proportional to the structural scale of the particle-based deformable surface. According to remark 9 and example 3, the Jacobian matrix $\overline{\mathbf{J}}_{sb}$ is a block-diagonal matrix, with the maximum dimension 4×4 for each block. So regardless of the total dimension of the matrix $\overline{\mathbf{J}}_{sb}$, it is always possible to break it into block diagonals with appropriate dimensions in order to make the inverse calculations according to the timing constraints of a real time simulation.

4.2 Simulation Results

For the simulation, the example of a small deformable surface modeled by 8 particles in a distributed fashion is considered in figure 17. As it can be seen in the figure, the two local nodes are identified by two distinct squares connected together by the two boundary algebraic equations.

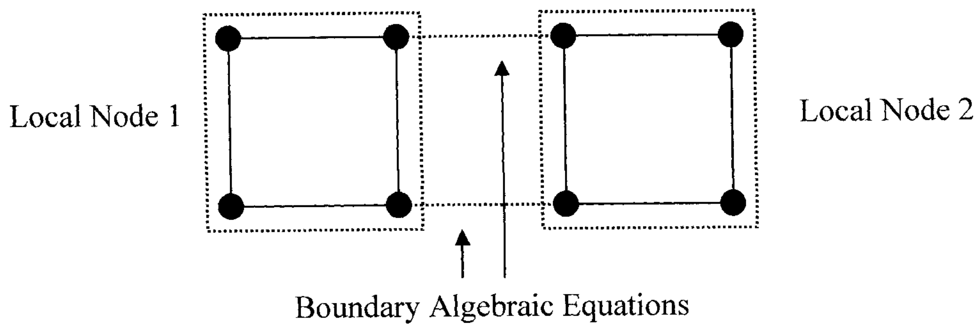


Figure 17. The 8-particle distributed model of a small deformable surface.

In this figure, the spring-damper forces between the particles are not shown. The spring-damper structure (figure 15) of the example above is shown in the following figure 18.

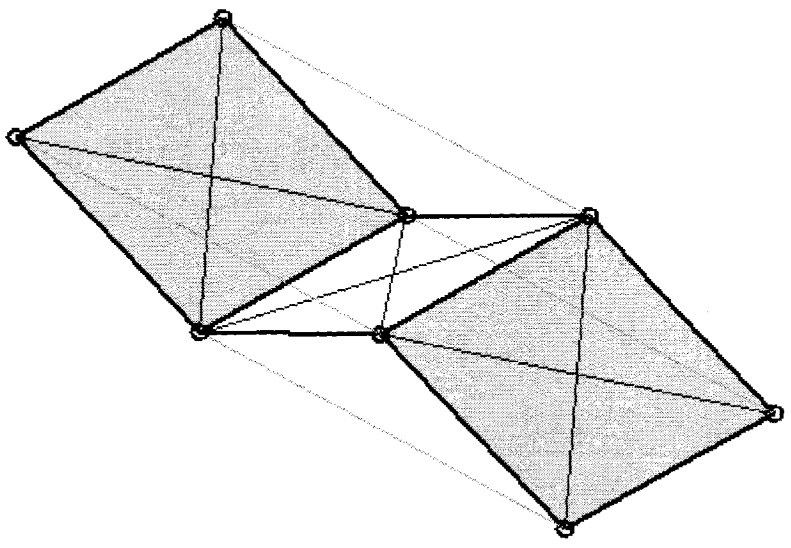


Figure 18. The spring-damper structure of the 8-particle distributed model of a small deformable surface.

In this figure, the thin lines represent the spring-damper force. Like before, the two distinct squares are the local nodes which are connected by two boundary algebraic equations. Each local node is simulated on a separate computer, and the inverse calculation of the corresponding local Jacobian matrix $\bar{\mathbf{J}}_{sk}$ is done on that computer. The inverse calculation of the Jacobian matrix $\bar{\mathbf{J}}_{sb}$ of the set of boundary algebraic equations is done on a separate computer as well.

The simulation is run in two steps, while the parameter values used in the simulation are as follows:

Masses	$m_1 = m_2 = m_3 = 0.2\text{Kg}$
Desired lengths	$l_1 = l_b = l_2 = 1\text{m}$
Sliding parameters	$\mu = 0.02$, $k = 20$
Simulation step size	$T = 0.001\text{Sec}$
Boundary layer thickness	$\varepsilon = 0.1$

- First step, the simulation is run for the inconsistent initial condition in figure 19. This structure is not subject to any external forces as gravity and wind.

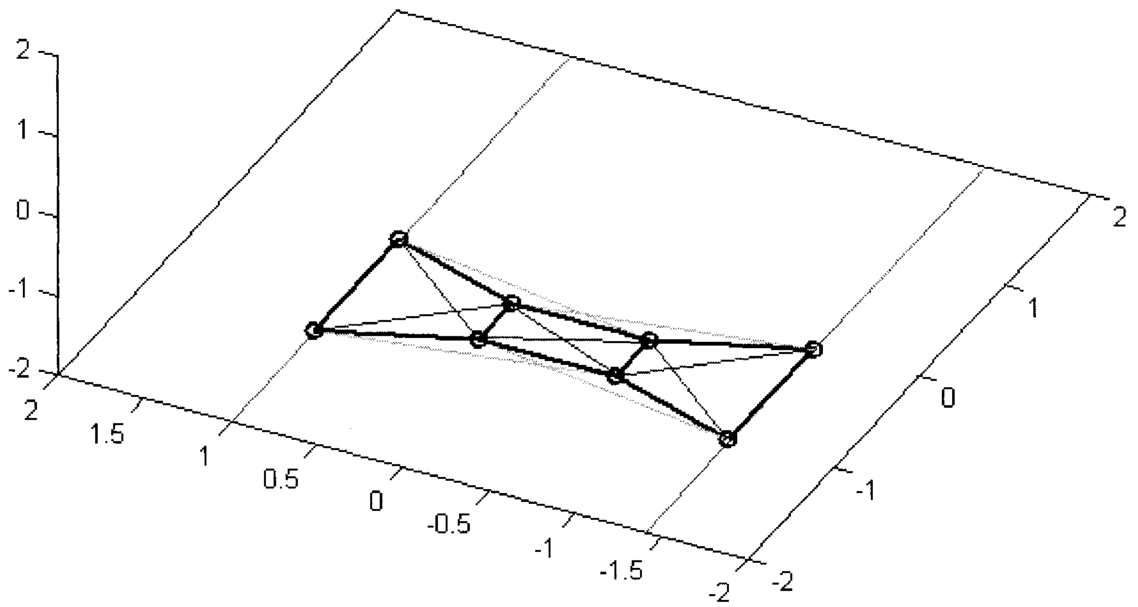


Figure 19. The inconsistent initial condition of the 8-particle distributed model of the deformable surface

Starting with this initial condition and after a finite reaching time, the 8-particle distributed model of the deformable surface converges to the stable formation shown in figure 20.

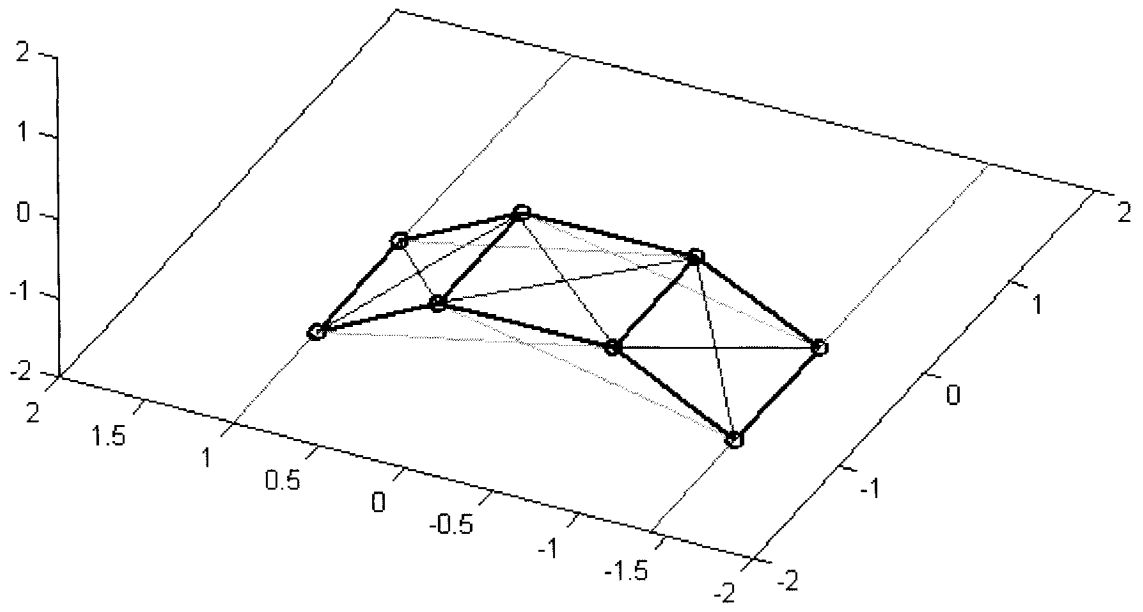


Figure 20. The stable formation, which the 8-particle distributed model converges to after a finite reaching time.

- Next step, the simulation is run for the consistent initial condition in figure 21. This structure is subject to the external forces as gravity and wind.

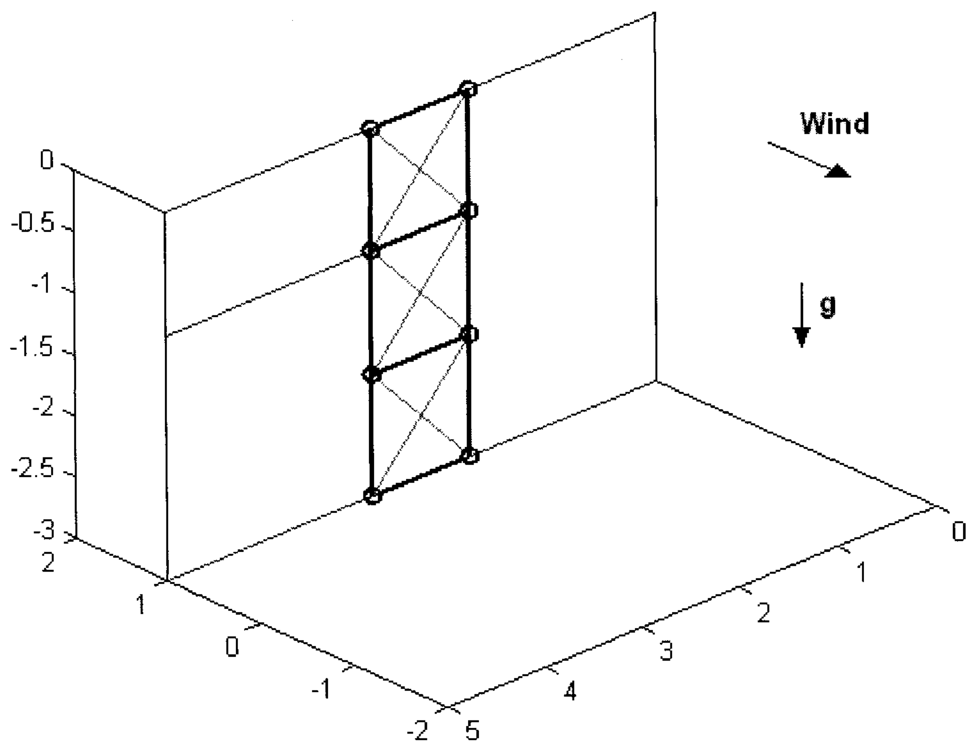


Figure 21. The consistent initial condition of the 8-particle distributed model of the deformable surface

Starting with this initial condition and after a finite reaching time, the 8-particle distributed model of the deformable surface takes the consistent formation shown in figure 22.

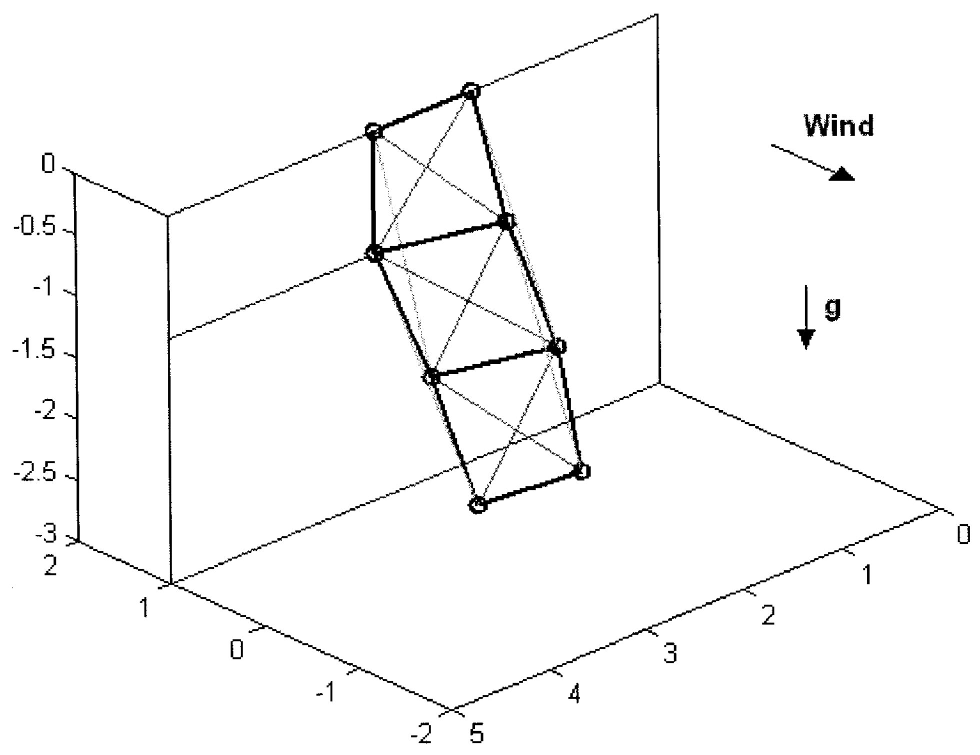


Figure 22. The consistent formation, which the 8-particle distributed model converges to after a finite reaching time.

So far, the qualitative effectiveness of the distributed method is verified by the simulation in two steps. In order to investigate the quantitative effectiveness of the method, the following graphs are presented. All these graphs pertain to simulation in the first step above.

Considering the reaching phase in the decentralized control (distributed simulation), from the initial state in figure 19 into the stable state in figure 20, the following 4 figures are related to one of the local rigid links in the first local node. Figure 23 is the local control input v_i of the local rigid link.

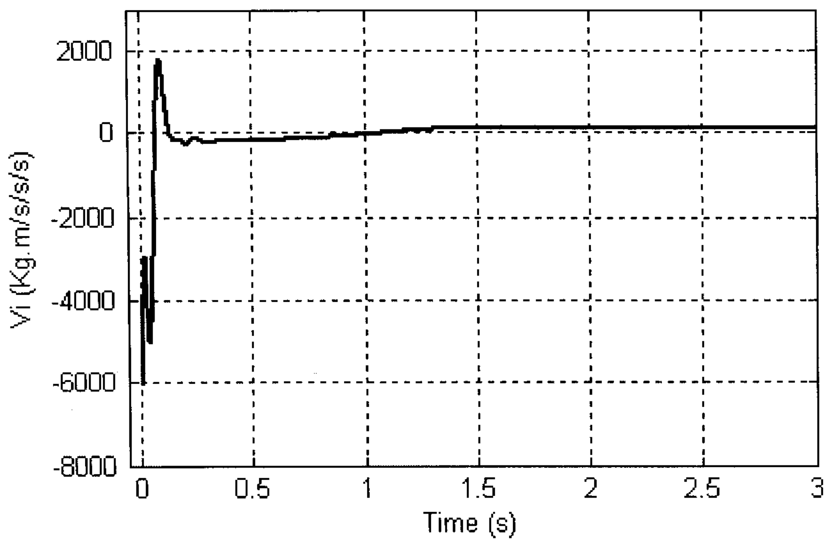
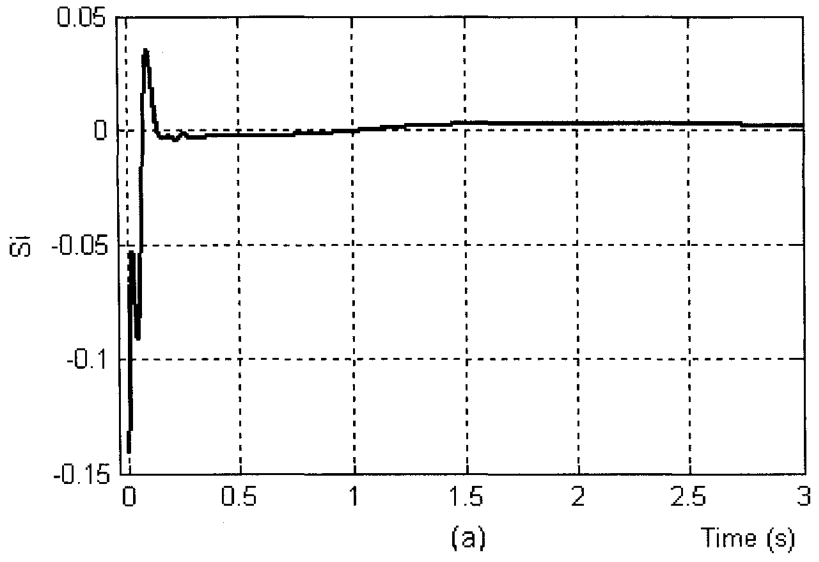


Figure 23. Decentralized control (distributed realization): the local control input v_i of the local rigid link.

The graph in figure 24-(a) shows the local sliding surface s_i corresponding to the local rigid link above.



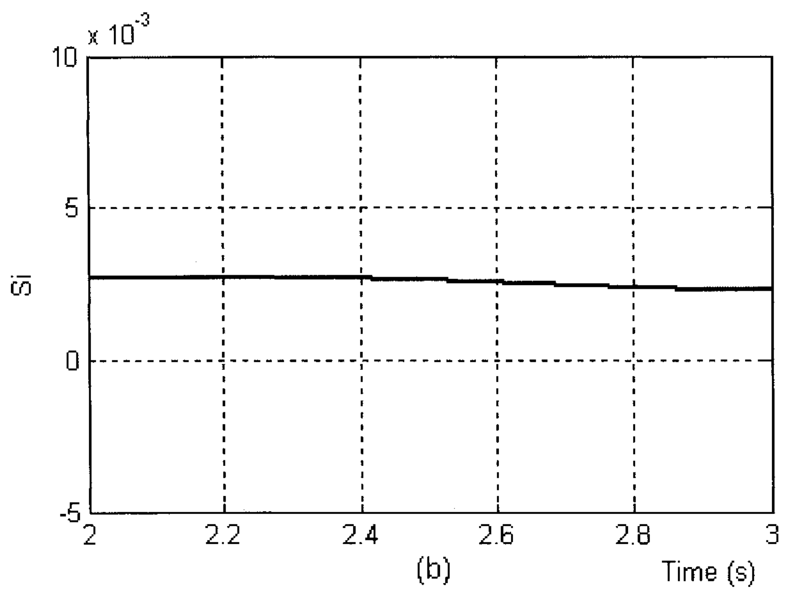


Figure 24. Decentralized control (distributed realization): the local sliding surface s_i corresponding to the local rigid link: (a) 0-3 sec, (b) zoom on 2-3 sec.

It shows that it goes to zero in less than $0.5\,(\text{sec})$. Figure 24-(b) shows that the error is less than 0.003 after 3 seconds. Hence, the local output w_i corresponding to the local rigid link above goes to zero in less than $0.5\,(\text{sec})$ as in figure 25-(a). Figure 25-(b) shows that the error is less than 0.003 after 3 seconds.

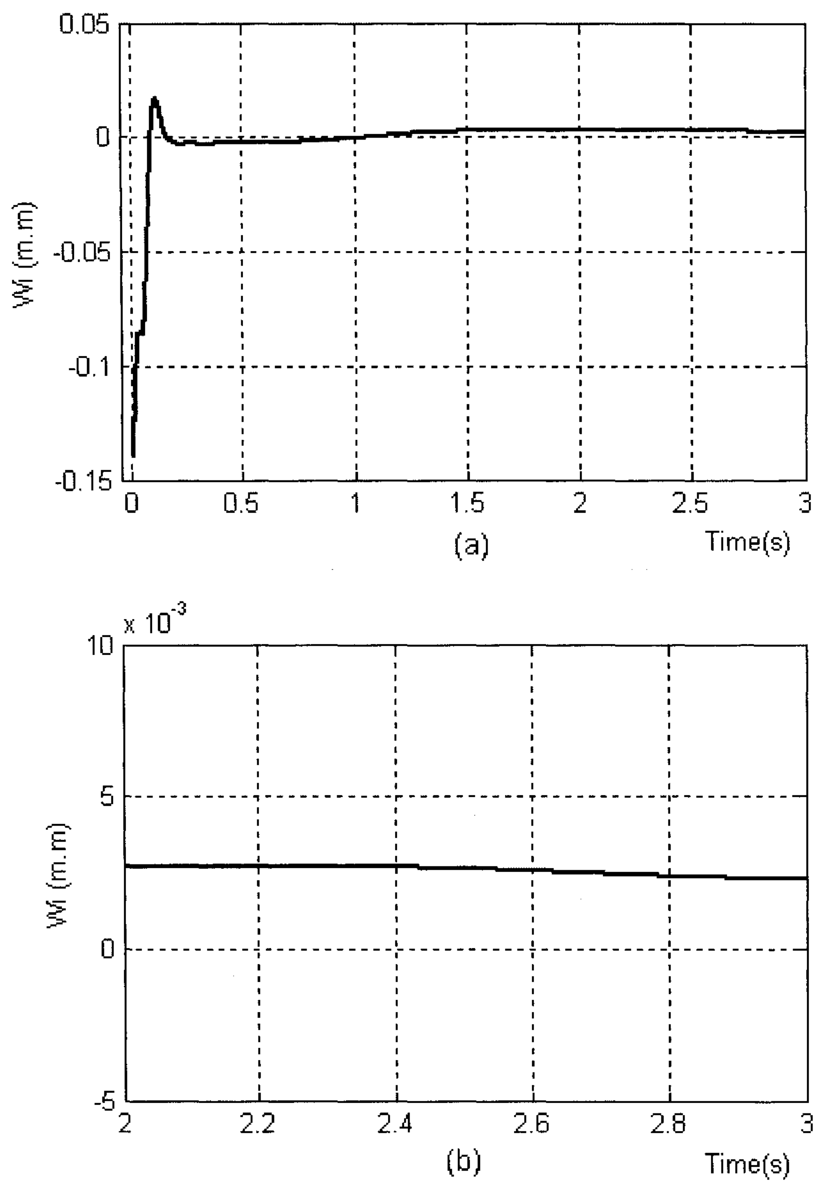


Figure 25. Decentralized control (distributed realization): the local output w_i corresponding to the local rigid link: (a) 0-3 sec, (b) zoom on 2-3 sec.

As a result, the length l_i of the corresponding local rigid link converges to the desired length $1\,(\mathrm{m})$ in less than $0.5\,(\mathrm{sec})$ as shown in figure 26.

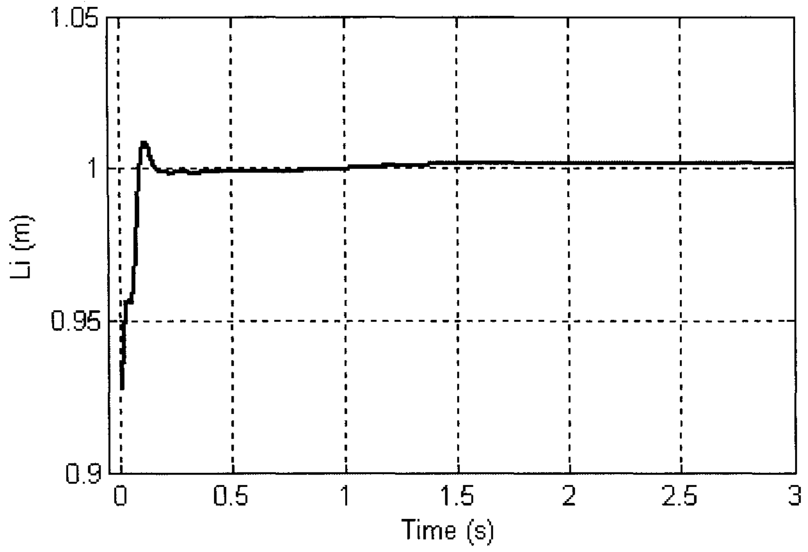


Figure 26. Decentralized control (distributed realization): the length l_i corresponding to the local rigid link.

Now, the following 4 figures are related to one of the boundary rigid links in the boundary node. Figure 27 is the boundary control input v_{bj} of the boundary rigid link.

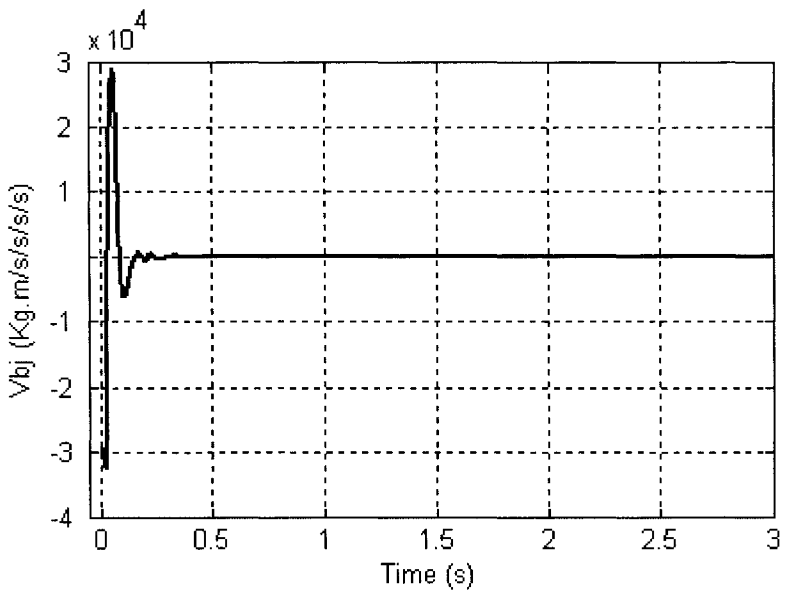


Figure 27. Decentralized control (distributed realization): the boundary control input v_{bj} of the boundary rigid link.

The graph in figure 28-(a) shows the boundary sliding surface s_{bj} corresponding to the boundary rigid link above.

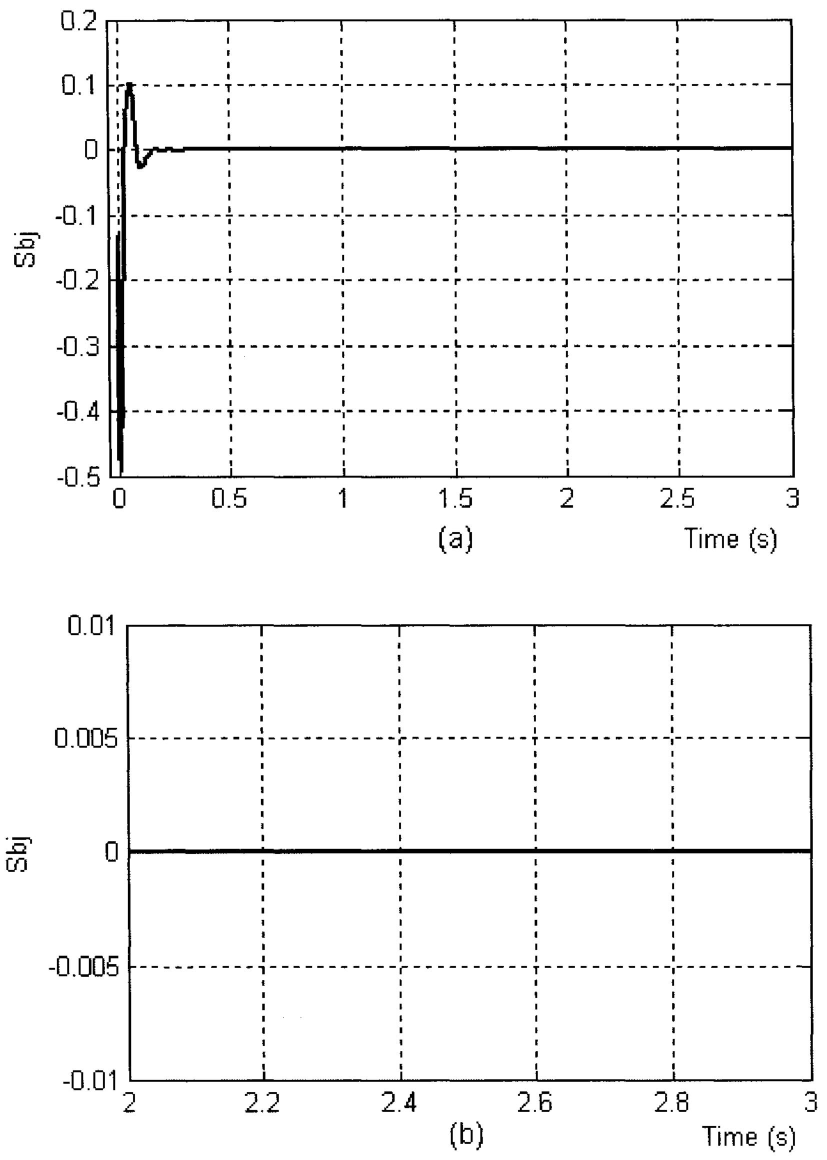


Figure 28. Decentralized control (distributed realization): the boundary sliding surface s_{bj} corresponding to the boundary rigid link: (a) 0-3 sec, (b) zoom on 2-3 sec.

It shows that it goes to zero in less than $0.5\,(\mathrm{sec})$. Figure 28-(b) shows that the error is almost zero after 3 seconds. Hence, the boundary output w_{bj} corresponding to the boundary rigid link above goes to zero in less than $0.5\,(\mathrm{sec})$ as in figure 29-(a).

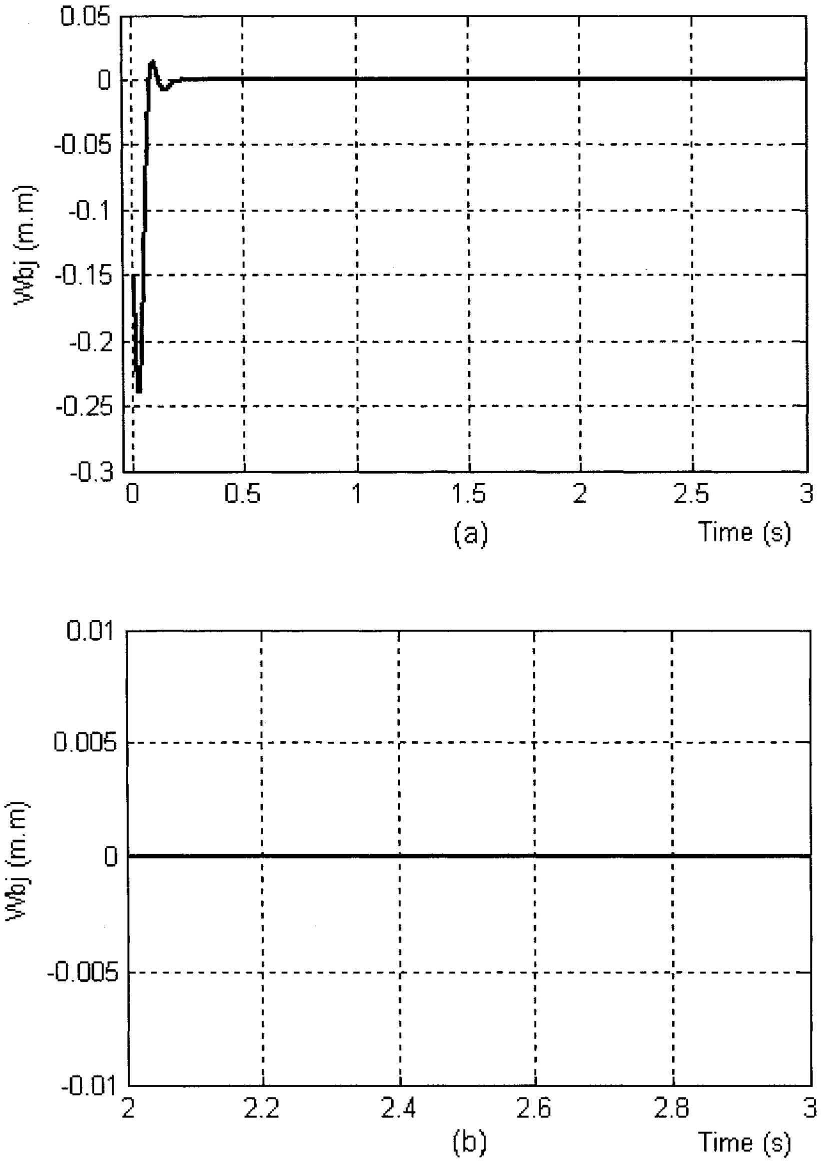


Figure 29. Decentralized control (distributed realization): the boundary output w_{bj} corresponding to the boundary rigid link: (a) 0-3 sec, (b) zoom on 2-3 sec.

Figure 29-(b) shows that the error is almost zero after 3 seconds. As a result, the length l_{bj} of the corresponding boundary rigid link converges to the desired length $1\,(\mathrm{m})$ in less than $0.5\,(\mathrm{sec})$ as shown in figure 30.

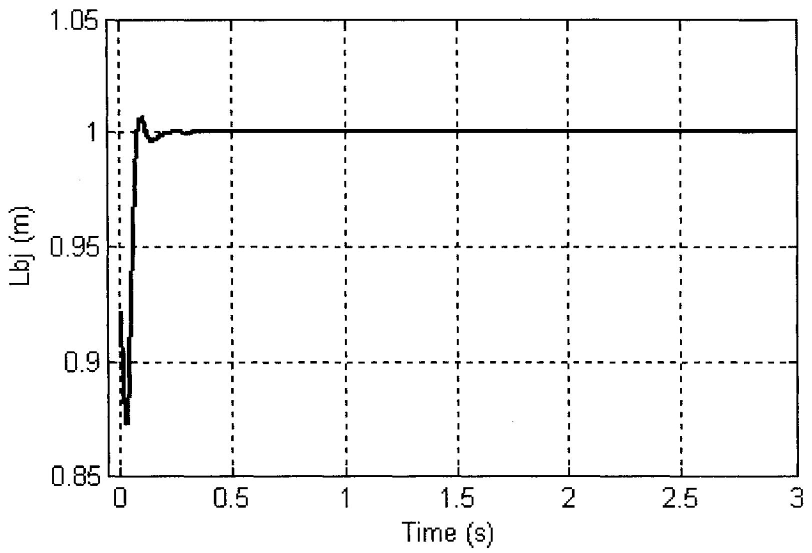


Figure 30. Decentralized control (distributed realization): the length l_{bj} corresponding to the boundary rigid link.

For comparison purposes, the same reaching phase, from the initial state in figure 19 into the stable state in figure 20, is simulated by the centralized controller. The following 4 figures are related to one of the local rigid links in the first local node. Figure 31 is the local control input v_i of the local rigid link.

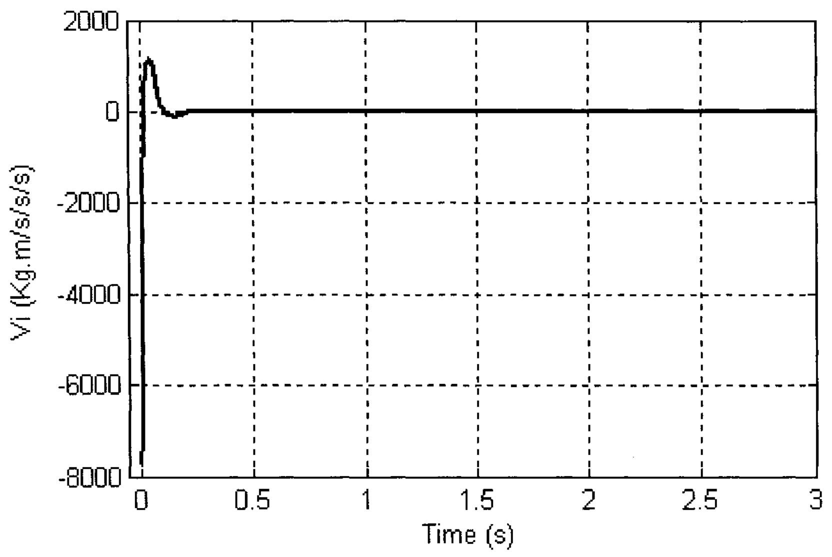


Figure 31. Centralized control: the local control input v_i of the local rigid link.

The graph in figure 32-(a) shows the local sliding surface s_i corresponding to the local rigid link above.

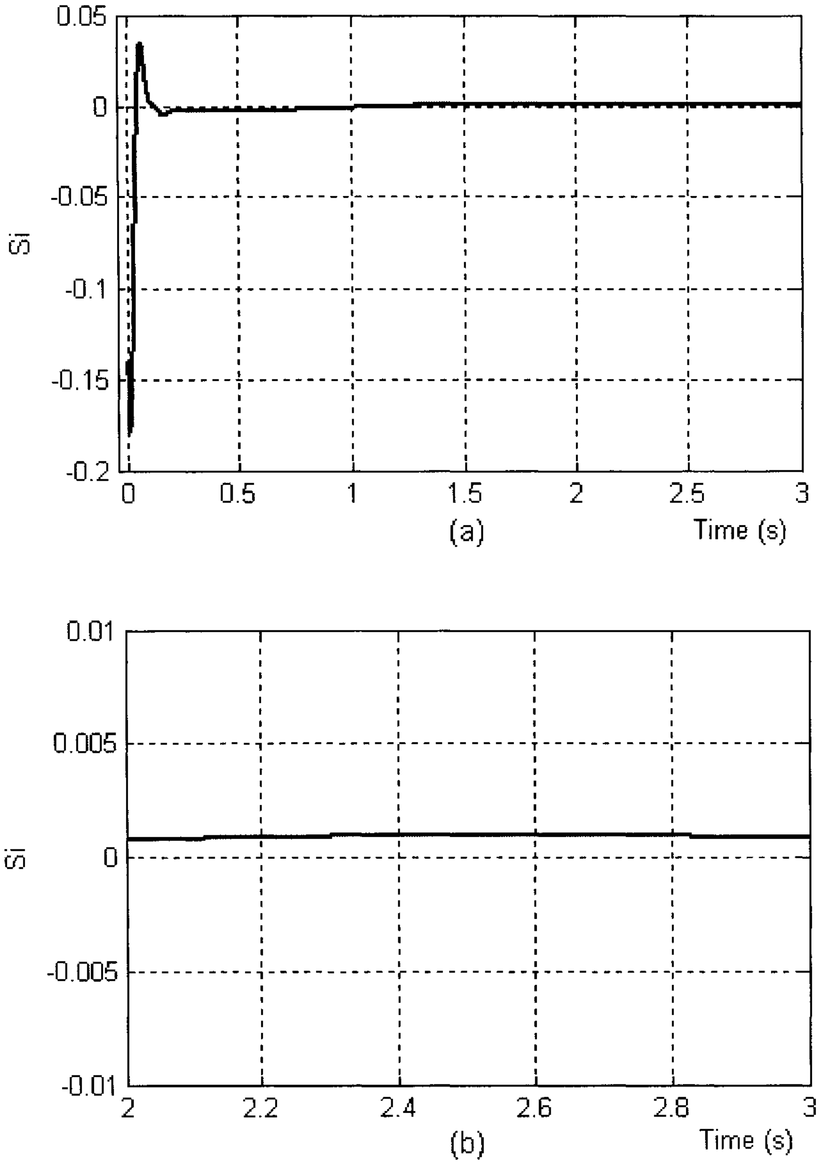


Figure 32. Centralized control: the local sliding surface s_i corresponding to the local rigid link: (a) 0-3 sec, (b) zoom on 2-3 sec.

It shows that it goes to zero in less than $0.5\,(\text{sec})$. Figure 32-(b) shows that the error is less than 0.001 after 3 seconds. Hence, the local output w_i corresponding to the local rigid link above goes to zero in less than $0.5\,(\text{sec})$ as in figure 33-(a).

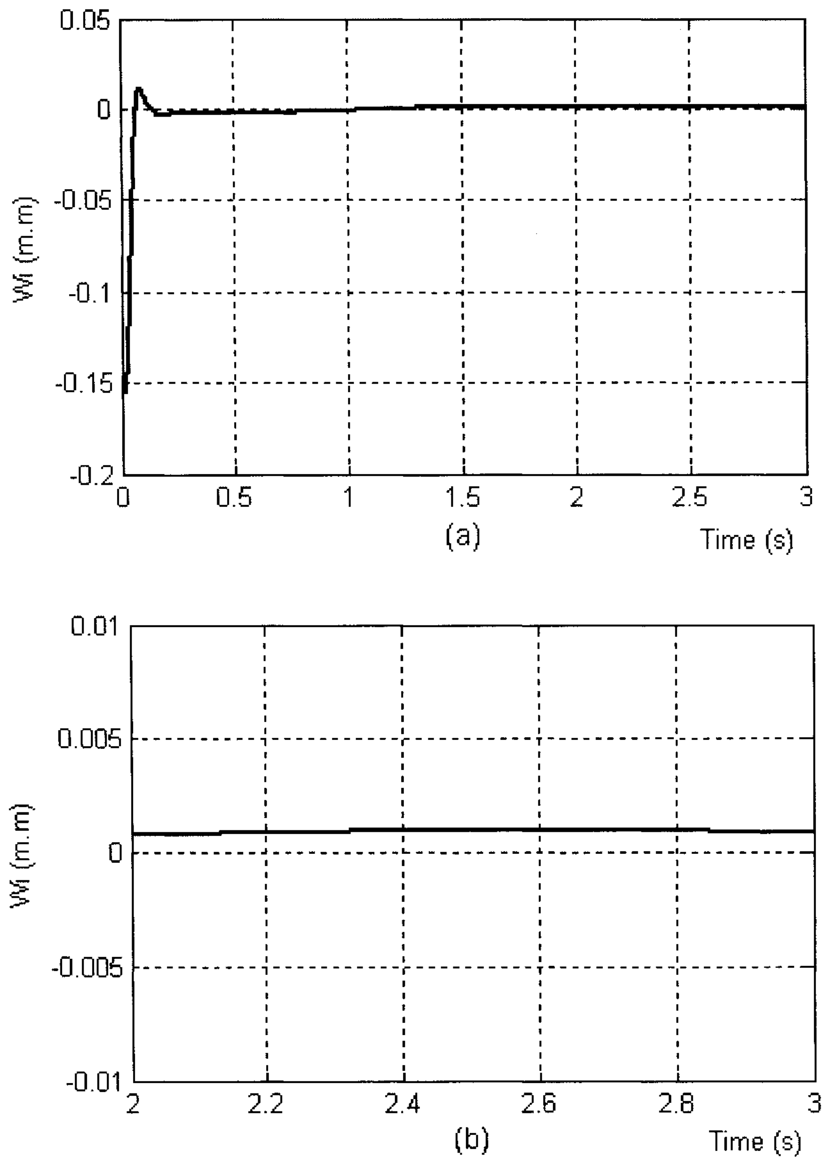


Figure 33. Centralized control: the local output w_i corresponding to the local rigid link: (a) 0-3 sec, (b) zoom on 2-3 sec.

Figure 33-(b) shows that the error is less than 0.001 after 3 seconds. As a result, the length l_i of the corresponding local rigid link converges to the desired length $1(\mathrm{m})$ in less than $0.5(\mathrm{sec})$ as shown in figure 34.

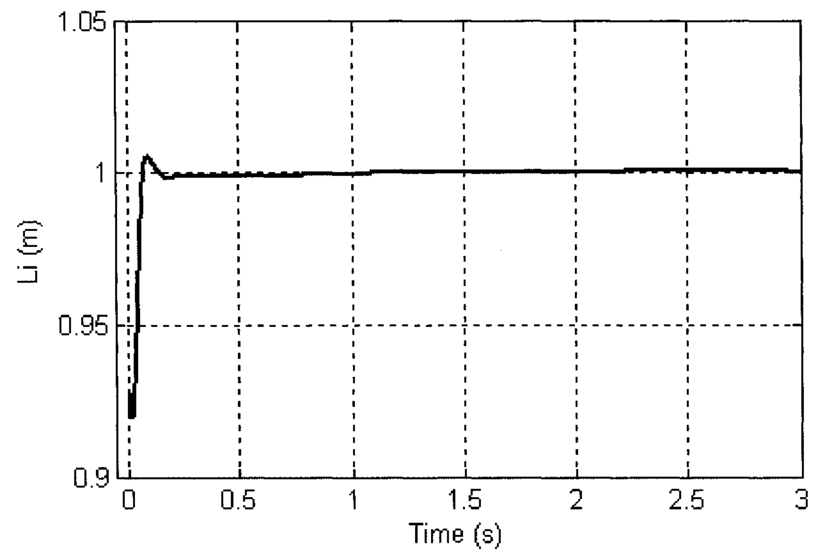


Figure 34. Centralized control: the length l_i corresponding to the local rigid link.

Comparing these four figures with the figures related to the decentralized case, it is obvious that the performance of the decentralized control in the distributed realization is quite similar to the centralized case, which confirms the validity of the analytical work. The only difference is the input v_{bj}, which has a larger order (10 times) in the decentralized case than the centralized one.

For the decentralized case, the Jacobian matrices of the two local nodes are shown in equation (107) below,

$$\bar{\mathbf{J}}_{si} = \begin{bmatrix} \bar{J}_{si(1,1)} & \bar{J}_{si(1,2)} & \bar{J}_{si(1,3)} & 0 \\ \bar{J}_{si(2,1)} & \bar{J}_{si(2,2)} & 0 & \bar{J}_{si(2,4)} \\ \bar{J}_{si(3,1)} & 0 & \bar{J}_{si(3,3)} & \bar{J}_{si(3,4)} \\ 0 & \bar{J}_{si(4,2)} & \bar{J}_{si(4,3)} & \bar{J}_{si(4,4)} \end{bmatrix}, \quad i = 1,2 \qquad (107)$$

while the Jacobian matrix of the boundary node is shown in equation (108) as well.

$$\bar{\mathbf{J}}_{sb} = \begin{bmatrix} \bar{J}_{sb(1,1)} & 0 \\ 0 & \bar{J}_{sb(2,2)} \end{bmatrix} \tag{108}$$

For the centralized case, the Jacobian matrix of the overall system is shown in equation (109) below.

$$\mathbf{J}_s = \begin{bmatrix}
J_{s(1,1)} & J_{s(1,2)} & J_{s(1,3)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
J_{s(2,1)} & J_{s(2,2)} & 0 & J_{s(2,4)} & J_{s(2,5)} & 0 & 0 & 0 & 0 & 0 \\
J_{s(3,1)} & 0 & J_{s(3,3)} & J_{s(3,4)} & 0 & J_{s(3,6)} & 0 & 0 & 0 & 0 \\
0 & J_{s(4,2)} & J_{s(4,3)} & J_{s(4,4)} & J_{s(4,5)} & J_{s(4,6)} & 0 & 0 & 0 & 0 \\
0 & J_{s(5,2)} & 0 & J_{s(5,4)} & J_{s(5,5)} & 0 & J_{s(5,7)} & J_{s(5,8)} & 0 & 0 \\
0 & 0 & J_{s(6,3)} & J_{s(6,4)} & 0 & J_{s(6,6)} & J_{s(6,7)} & 0 & J_{s(6,9)} & 0 \\
0 & 0 & 0 & 0 & J_{s(7,5)} & J_{s(7,6)} & J_{s(7,7)} & J_{s(7,8)} & J_{s(7,9)} & 0 \\
0 & 0 & 0 & 0 & J_{s(8,5)} & 0 & J_{s(8,7)} & J_{s(8,8)} & 0 & J_{s(8,10)} \\
0 & 0 & 0 & 0 & 0 & J_{s(9,6)} & J_{s(9,7)} & 0 & J_{s(9,9)} & J_{s(9,10)} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & J_{s(10,8)} & J_{s(10,9)} & J_{s(10,10)}
\end{bmatrix} \tag{109}$$

Comparing the Jacobian matrices above, the effectiveness of the distributed simulation in calculating the inverse of the Jacobian matrix is clear. Specifically, when we are dealing with a large scale system with a considerably large dimension Jacobain matrix, the distributed simulation saves enough time to meet the timing restrictions of a real time simulation.

The other important point is that for both the centralized and decentralized cases, the maximum simulation step size is 0.008 (sec). It means that by using the decentralized control (distributed simulation) instead of centralized one, there is no considerable stiffness added to the system, which is the main objective of the distributed simulation.

5 . Conclusions and Future Work

5.1 Conclusions

The sliding control approach is a quite efficient method to simulate the mechanical DAE systems. The most significant problem is when the dimension of the algebraic equations increases considerably. In this case, the dimension of the Jacobian matrix increases accordingly. As a result, computing the inverse of the Jacobian matrix becomes too expensive for a single computer, specifically when real time simulation is required. The proposed approach is to use a network of computers instead of a sigle one.

To implement the simulation of dynamic systems on a computer network, a method based on decentralized sliding mode control was presented in this work. This method divided the main system into numerous nodes. A sliding mode controller was designed for each node (including the local nodes and boundary node), individually. The necessary conditions for stability were developed. The new method was applied to animation of a deformable surface which demonstrated good performance and stability. The main contribution of the thesis could be listed as:

- Remodeling the DAE system and transforming it into the distributed formation.
- Applying the decentralized sliding controller method.

Simulation results confirmed the efficiency and validity of the analytical work. The advantageous of decentralized (distributed) method could be highlighted as:

- The performance of the decentralized (distributed) method is quite similar to the centralized one.
- In the decentralized (distributed) method, we are dealing with several Jacobian matrices whose dimensions are considerably less than the dimension of the single Jacobian matrix in the centralized case. Therefore, the distributed approach fits better to the timing restrictions of a real time simulation.
- In the decentralized (distributed) method, the maximum simulation step size is not much different from the one in the centralized case. This shows that the distributed method does not add much stiffness to the system.

Therefore, the distributed control approach proposed is an effective solution to use a number of computers instead of a single PC, in order to meet the timing requirements of a real-time simulation.

5.2 Future Works

In this section, the possible future directions for the approach proposed in this thesis are stated. These possible directions include implementing the observer-based decentralized sliding control methods [18,21,22,30,37]. Moreover, better estimation methods based on adaptive neuro-fuzzy networks can be applied to estimate the unknown nonlinear interconnections, disturbances and unmodelled dynamic uncertainties. Also, the current flow chart can be replaced by a more efficient and systematic algorithm to design the sliding control parameters. For the simulation section, a real computer nework can be implemented with the proper local programs including the necessary algorithms, so that the real time simulation of a large scale deformable surface can be done on-line.

6 . References

[1] Luenberger, D.G., "Dynamic Equations in Descriptor Form", *IEEE Transactions on Automatic Control*, vol. ac-22, no. 3, June 1977.

[2] Gordon, B.W., and Liu S., "A Singular Perturbation Approach for Modeling Differential-Algebraic Systems", *ASME Journal of Dynamic Systems, Measurement, and Control*, December 1998.

[3] Gordon, B.W., and Asada, H., "Modeling, realization, and simulation of thermo-fluid systems using singularly perturbed sliding manifolds," *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 122, no. 4, pp. 699-707, 2000.

[4] Gordon, B.W., *State Space Modeling of Differential-Algebraic Systems using Singularly Perturbed Sliding Manifolds*, Ph.D. Thesis, MIT, Mechanical Engineering Dept., 1999.

[5] Slotine, J.-J.E, "Sliding Controller Design for Nonlinear Systems", *International Journal of Control,* vol.40, no.2, 1984.

[6] Chiang-Cheng Chiang; Zu-Hung Kuo; Tzu-Ching Tung, "Decentralized adaptive fuzzy sliding-mode controller design for large-scale nonlinear uncertain systems", *The 10th IEEE International Conference on Fuzzy Systems*, vol: 2, pp: 732 – 735, Dec. 2001.

[7] L. Shi and S. K. Singh, "Decentralized adaptive controller design for large-scale systems with higher order interconnections," *IEEE Transactions on Automatic Control*, vol. 37, no. 8, pp. 1106-1118, Aug., 1992.

[8] Hsu, K.C., "Decentralized variable structure model-following adaptive control for interconnected systems with series nonlinearities", *International Journal of Control*, vol: 29, pp: 365 – 372, Apr. 1998.

[9] Zhang, T.P., and Feng, C.B., "Decentralized adaptive fuzzy control for large-scale nonlinear systems," *Fuzzy Sets and Systems,* vol. 92, pp. 61-70, Nov. 1997.

[10] Wang, W.J., and Lee, J.L., "Decentralized variable structure control design in perturbed nonlinear systems", *ASME Journal of Dynamic Systems, measurement, and Control*, vol.115, pp. 551-554, 1993.

[11] Brenan, K., Campbell, S. and L. Petzold, *Numerical Solution of Initial Value Problems in Differential-Algebraic Equations*, North-Holland, Amsterdam, 1989

[12] Feipeng Da, "Decentralized sliding mode adaptive controller design based on fuzzy neural networks for interconnected uncertain nonlinear systems", *IEEE Transactions on Neural Networks*, Volume 11, Issue 6, Nov. 2000 Page(s):1471 – 1480.

[13] Chiang-Cheng Chiang; Zu-Hung Kuo; Tzu-Ching Tung, "Decentralized adaptive fuzzy sliding-mode controller design for large-scale nonlinear uncertain systems", *The 10th IEEE International Conference on Fuzzy Systems*, Vol. 2, 2-5 Dec. 2001, Page(s):732 – 735.

[14] Shi, L.; Singh, S.K., "Decentralized adaptive controller design for large-scale systems with higher order interconnections", *IEEE Transactions on Automatic Control*, Volume 37, Issue 8, Aug. 1992, Page(s):1106 – 1118.

[15] Cunchen Gao; Yongqing Liu, "On the design of decentralized variable structure controllers of neutral nonlinear large-scale control systems with delays", *IEEE*

International Conference on Systems, Man, and Cybernetics, Volume 1, 14-17 Oct. 1996, Page(s):292 – 297.

[16] Chien-Hsin Chou; Chih-Chiang Cheng, "Decentralized model following variable structure control for perturbed large-scale systems with time-delay interconnections", *Proceedings of the American Control Conference*, Volume 1, Issue 6, 28-30 June 2000, Page(s):641 – 645.

[17] Morgan, R.; Ozguner, U., "A decentralized variable structure control algorithm for robotic manipulators", *IEEE Journal of Robotics and Automation* [legacy, pre - 1988], Volume 1, Issue 1, Mar 1985, Page(s):57 – 65.

[18] Wu, Q. H.; Jiang, L.; Wen, J. Y., "Decentralized Adaptive Control of Interconnected Non-Linear Systems Using High Gain Observer", *International Journal of Control*, 5/20/2004, Vol. 77, Issue 8, p703-712.

[19] Lin, W. S. and Chen, C. S., "Robust Adaptive Sliding Mode Control Using Fuzzy Modeling for a Class of Uncertain MIMO Nonlinear Systems", *IEE Proceedings of Control Theory & Applications*, Jul2004, Vol. 151, Issue 4, p522-524.

[20] Ha, Q. P.; Rye, D. C.; Durrant-Whyte, H. F., "Robust sliding mode control with application", *International Journal of Control*, 08/15/99, Vol. 72, Issue 12, p1087.

[21] Gow-Bin Wang, Sheng-Shiou Peng and Hsiao-Ping Huang, "A sliding observer for nonlinear process control", *Chemical Engineering Science*, Volume 52, Issue 5, March 1997, Pages 787-805.

[22] Jacques Hernandez and Jean-Pierre Barbot, "Sliding observer-based feedback

control for flexible joints manipulator, *Automatica*, Volume 32, Issue 9, September 1996, Pages 1243-1254.

[23] Jinyong Yu; Wenjin Gu; Youan Zhang, "New adaptive fuzzy sliding-mode control system for SISO nonlinear system", *Fifth World Congress on Intelligent Control and Automation*, WCICA 2004, Volume 2, 15-19 June 2004, Page(s):1196 – 1199.

[24] Jinyong Yu; Daquan Tang; Wenjin Gu; Yigao Deng, "Adaptive fuzzy sliding-mode controller for nonlinear system with a general set of uncertainty", *Control, Automation, Robotics and Vision Conference*, ICARCV 2004 8th, Volume 3, 6-9 Dec. 2004, Page(s):1912 – 1916.

[25] Feipeng Da; Shumin Fei; Xianzhong Dai, "Sliding mode adaptive output feedback control of nonlinear systems using neural networks, *Proceedings of American Control Conference*, June 8-10, 2005, Page(s):1721 - 1726.

[26] Chen, Jen-Yang, "Fuzzy Sliding Mode Controller Design: Indirect Adaptive Approach", *Cybernetics & Systems*, Jan1999, Vol. 30, Issue 1, p9-27.

[27] Yu, Xinghuo; Zhihong, Man., "Fuzzy Sliding Mode Control Systems with Adaptive Estimation", *Cybernetics & Systems*, Oct99, Vol. 30, Issue 7, p663-680.

[28] Haojian Xu; Mirmirani, M.; Ioannou, P.A.; Boussalis, H.R., "Robust adaptive sliding control of linearizable systems", *Proceedings of American Control Conference*, Volume 6, 25-27 June 2001, Page(s):4351 – 4356.

[29] Byungkook Yoo; Woonchul Ham, "Adaptive fuzzy sliding mode control of nonlinear system", *IEEE Transactions on Fuzzy Systems*, Volume 6, Issue

2, May 1998, Page(s):315 – 321.

[30] Jiang, L.; Wu, Q.H., “Nonlinear adaptive control via sliding-mode state and perturbation observer”, *Control Theory and Applications*, Volume 149, Issue 4, July 2002, Page(s):269 – 277.

[31] Shuanghe Yu; Xinghuo Yu; Efe, M.O., “Modeling-error based adaptive fuzzy sliding mode control for trajectory-tracking of nonlinear systems”, *IEEE Industrial Electronics Society IECON*, Volume 3, 2-6 Nov. 2003, Page(s):3001 – 3006.

[32] Su, Juhng Perng; Huang, Ching Ting, “Sigma adaptive fuzzy sliding mode control of a class of nonlinear systems”, *International Journal of Systems Science*, Aug2000, Vol. 31 Issue 8, p949-959.

[33] Chen, Jen-Yang, “Design of Adaptive Fuzzy Sliding Mode Control for Nonlinear Systems”, *International Journal of Uncertainty, Fuzziness & Knowledge-Based Systems*, Oct99, Vol. 7, Issue 5, p463.

[34] Shaocheng Tong; Han-Xiong Li, “Fuzzy adaptive sliding-mode control for MIMO nonlinear systems”, *IEEE Transactions on Fuzzy Systems*, Volume 11, Issue 3, June 2003, Page(s):354 – 360.

[35] Chiang-Cheng Chiang; Yan-Shiang Tzeng, “Dynamic sliding mode adaptive controller for nonlinear systems with higher-order and unmatched uncertainties, *IEEE International Conference on Systems, Man, and Cybernetics*, Volume 1, 12-15 Oct. 1999, Page(s):44 – 49.

[36] Huang, An-Chyau; Kuo, Yeu-Shun, “Sliding control of non-linear systems containing time-varying uncertainties with unknown bounds”, *International*

Journal of Control, 02/15/2001, Vol. 74, Issue 3, p252-264.

[37] Chung-Chun Kung and Ti-Hung Chen, “Observer-based indirect adaptive fuzzy sliding mode control with state variable filters for unknown nonlinear dynamical systems”, *Fuzzy Sets and Systems*, Volume 155, Issue 2, 16 October 2005, Pages 292-308.

[38] Sastry, S.S.; Isidori, A., “Adaptive control of linearizable systems”, *IEEE Transactions on Automatic Control*, Volume 34, Issue 11, Nov. 1989, Page(s):1123 – 1131.

[39] House, D., Breen, D. E., (ed.), *Cloth Modeling and Animation*. A.K. Peters, Natick Mass., c2000.

[40] Provot, X., “Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behaviour”, *Proceeding of Graphics Interface*, vol. 95, pp. 147-154, 1995.

[41] Desbrun, M., Meyer, M., Barr, A.H., “Interactive Animation of Cloth-Like Objects for Virtual Reality”, *Journal of Vizualisation and Computer Animation*, 2000.

[42] Ehrhardt, K., Borchardt, J., Grund, F., Horn, D., “Distributed dynamic process simulation”, *ZAMM*, vol. 81, Suppl. 3, pp. S715-S716, 2001.

[43] Borchardt, J., “Newton-type decomposition methods in parallel process simulation of chemical plants”, *AIDIC Conference Series*, vol. 5, pp. 57-64, 2002.

[44] Borchardt, J., “Newton-type decomposition methods in large-scale dynamic process simulation”, *Computers and Chemical Engineering*, vol. 25, pp. 951-961, 2001.

[45] Borchardt, J., Grund, F., Horn, D., “Parallelized methods for large nonlinear and linear systems in the dynamic simulation of industrial applications”, *Surveys on Mathematics for Industry*, vol. 8, pp. 201-211, 1999.

[46] Ehrhardt, K., Borchardt, J., Grund, F., Horn, D., “Divide and conquer strategies in large scale dynamic process simulation”, *Computers and Chemical Engineering*, vol. 23, Suppl., pp. S335-S338, 1999.

[47] Wang, J., Gosselin, C.M. and Cheng, L., “Dynamic modeling and simulation of parallel mechanisms using the virtual spring approach”, *Proceedings of ASME Design Engineering Technical Conferences*, Baltimore, Maryland, 2000.

[48] Lee, C. S. G. and Chang, P. R., “Efficient parallel algorithms for robot inverse dynamics computation”, *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 16, pp. 532-542, 1986.

[49] Lee, C. S. G. and Chang, P. R., “Efficient parallel algorithms for robot forward dynamics computation,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 18, pp. 238-251, 1988.