In Proceedings of the 8th ESA Workshop on Advanced Space Technologies for Robotics and Automation
'ASTRA 2004' ESTEC, Noordwijk, The Netherlands, November 2 - 4, 2004

OBJECT-ORIENTED MODELING OF A SPACE ROBOTIC MANIPULATOR

Gianni Ferretti(1), GianAntonio Magnani(1), Paolo Rocco(1), Luca Viganò(1), Marco Gritti(1), Andrea Rusconi(2)
(1)Politecnico di Milano
Dipartimento di Elettronica e Informazione
Piazza L. da Vinci, 32, Milano, Italy
{ferretti, magnani, rocco, vigano}@elet.polimi.it

(2)Galileo Avionica
Via Montefeltro 8, Milano, Italy
andrea.rusconi@officine-galileo.finmeccanica.it

***Abstract*—In this paper a detailed mechatronic model of SPIDER, the Italian space manipulator arm, is presented. The model is based on the open source modeling language Modelica. Thanks to the distinctive features of this language, like object orientation, physical and acausal modeling, symbolic handling of equations and constraints, the overall model of SPIDER is obtained by assembling few models of its components, all written in a fairly natural way. Remarkably simple is the modeling of contact operations against a perfectly rigid environment.**

I. INTRODUCTION

Modeling and simulation of a space robot is a challenging task. Difficulties are related to the multi-body mechanical chain modeling, nonlinearities of friction model, impact and contact modeling, which even change the model structure in case of rigid environment. Furthermore, a robot is not just a mechanical structure: motors, power electronics and control laws concur to determine its performance and behavior in a synergic way.

Several powerful environments are available for space systems simulation. SYMOFROS [1], DCAP [2], ROSESAT [3], DARTS SHELL [4] are well known. In commercial, like ADAMS [5], and open source, like MBDYN [6], multi-body packages, the core is the multi-body dynamics of a mechanical chain, while less attention is given to simulation of joint and power electronics dynamics.

Experience on analysis and control of robots and machining centers [7] has pointed out that motion performance of position servomechanisms at low speed is adversely affected by torque disturbances, friction, and joint torsional flexibility. This is likely to happen also for space manipulation arms to be installed on the International Space Station (ISS), which must move very slowly, keeping acceleration very low, in order to avoid disturbances to microgravity condition. Actually, this is the case of the Italian space arm SPIDER/EUROPA, a 7 degrees of freedom (d.o.f.) manipulator to be installed on the EXPRESS Pallet platform on the International Space Station (ISS).

This paper presents a model of the SPIDER arm developed in the DYMOLA environment and written in the open source modeling language Modelica [8], [9]. Modelica is an object oriented language which comes with a set of multidomain libraries of models of elemental mechanical, electrical, electronics and control subsystem. Modelica offers multidomain physical modeling [10], acausal modeling, efficient symbolic handling of equations and constraints, which can be exploited to build the complex model of SPIDER from few models (modules) of its basic physical components, and including contact / non contact operations.

The model of the joints, made up of the motor, the gearbox, and the driver modules has been checked and tuned based on data available from an experiment carried out onboard the MIR station.

The paper is organized as follows. Section II discusses requirements and solutions for modeling and simulation of space robots. Section III gives details on the SPIDER manipulation arm and its model, and Section IV deals with joint model validation. Section V shows some simulation results.

II. OBJECT-ORIENTED APPROACH FOR SPACE ROBOT MODELING AND SIMULATION

As far as modeling approach and software tools are concerned, general requirements for space robot simulation can be classified as follows:

A. Multi Domain Simulation

Robots are mechatronic systems whose dynamic performances are affected by the mechanical, electrical and control subsystem, and their mutual interactions. Modeling a robot requires careful and balanced modeling of the most important phenomena in the three domains.

B. Modular Modeling and Software Reuse

Modular modeling and software reuse are obvious requirements for simulation of complex systems like space robots. However, modularization of mechanical systems is largely prevented in most simulation tools, which do not deal with systems where algebraic constraints are present. These constraints emerge from the connection of modules, for instance the rigid connection of the dynamic models of a motor shaft and of a gearbox.

C. Reliability and Efficiency of Numerical Simulation

Reliability and efficiency of the numerical solver are of major importance in space robot simulation. A DAE [11] (Differential Algebraic Equations) solver is needed, able to handle events, such as discontinuities and abrupt changes in external variables, as those occurring at impacts of end effector against the environment. The time instant when the event is triggered must be detected with precision, thanks to variable stepsize in the numerical simulation. Also, simulation of closed loop systems must be dealt with in the most reliable way.

D. Interfaces for Pre- and Post- Processing

Both 2D and 3D interfaces are used, the 2D being more common and easy to use. Each module must be associated with a graphical symbol as evocative of the physical nature of the system as possible. Visualization of simulation results can be performed both with 2D plots or 3D animation. 2D plots are more useful for detailed analysis of dynamic properties, while 3D animation helps understanding the gross motion of the system. Assembly of modules must be performed through a GUI.

The best answer to this set of requirements deeply involves the right choice of the simulation paradigm, the adoption of the object oriented approach to modeling, the availability of powerful and well tested libraries of basic models easily customizable to comply with specific application domains, and the efficient handling of equations and constraints.

A. ***Simulation Paradigm***

Declarative modeling languages should be used to simulate the physical parts of the system, be they mechanical or electrical components. Procedural languages, where the unidirectional flow of information from an input to an output is postulated, are inadequate to simulate systems where instantaneous exchanges of power are present. Acausal models, described by DAE systems, should be used to represent in the most natural and physically consistent way each system component, while the task of defining the computational causality of the assembled model should be in charge of the simulation environment.

B. Object Orientation

The modeling language should comply with modern object-oriented paradigms of software engineering. The most relevant aspects of object-orientation for physical system modeling are modularization (the system is composed by aggregation of modules), abstraction (the internal description of the module is separated from its interface), information encapsulation (only the interface variables are accessible to the other modules), reuse of the modules through parameterization. Structuring the modules in libraries greatly facilitates the use of the simulation environment for wider projects.

C. Customization

Every simulation environment comes with built-in libraries of components. These libraries usually cover the needs for simulation of simple mechatronic systems. However situations arise where customized modules must be written. An example is the model of a brushless motor, dealt with later on. In this respect, expansion of library modules, rather than writing customized modules from scratch, obviously facilitates the task of the user.

D. Efficient Handling of Equations and Constraints

Assembling of modules described by DAE equations generally leads to higher index systems and algebraic loops. Symbolic manipulation of the equations is then required to yield a set of equations amenable to numerical integration. Some simulation tools use Pantelides' [12] algorithm (or similar) to reduce the index of the system. Others use constraint relaxation techniques, generating an ODE system where, however, constraints are not identically satisfied during transients. The efficiency of this preprocessing software layer might be crucial for the efficiency of the simulation environment altogether.

The open source language Modelica and its commercial interpreter DYMOLA own these features, and have been profitably used in the derivation of the following model of the SPIDER manipulation arm.

III. THE MODEL OF THE SPIDER ARM

The SPIDER manipulation arm [13] (Fig. 1) was developed on behalf of Italian Space Agency (ASI) by a team of Italian companies led by Tecnospazio (now Galileo Avionica) in a project ended in 1998. A second version of the arm, with slight modifications, is currently under development as a part of the EUROPA (External Use of Robotics for Payload Automation) experiment, scheduled on the EXPRESS Pallet platform on the International Space Station (ISS).

Fig. 1.The Spider Arm

SPIDER is an anthropomorphic arm with 7 rotational d.o.f.. Each joint is powered by an electromechanical actuation group made up of a brushless motor, with brake and a resolver position sensor, and gearbox built around a Harmonic Drive transmission. In addition to the resolver on the motor shaft, a second resolver is placed on the output shaft of the gearbox, at the load side. Some distinctive features of the arm are the payload of 250 Kg (under 0 g conditions), the length of about 1670 mm, the mass of 58 Kg, a maximum linear speed of 0.1 m/s (limited by software), and a power consumption of 30-130 W. The end effector is equipped with tactile sensors on the two jaws in order to monitor the gripping force, and of a wrist force/torque sensor, for force/position control in contact motion.

The SPIDER arm consists of the following basic components: mechanical chain, two-phase brushless motors, drivers, gearboxes and controllers. Accurate models of these components, referred to as modules, have been obtained using the Modelica libraries as far as possible, and extending them when needed. Special attention has been given to modeling of torque disturbances, jointly due to motors and drivers, and of friction and torsional flexibility in the transmission gearbox, since they are most responsible of irregular motion at low speed in servomechanisms. Since modules are interfaced in exactly the same way as the physical components do (e.g., through electric terminals and mechanical flanges for the brushless motor), model readability and ease of assembling are strongly enhanced. Simulation of either elastic or rigid contact simply requires to add a specific module to the mechanical chain even if it imposes kinematic constraints on some state variables. While the mechanical chain model is specific to SPIDER, models of motors, drivers, and gearboxes are suitable to be used for simulation of generic space and industrial servomechanisms.

A. Mechanical Chain

The model of the 7 d.o.f mechanical chain of SPIDER is built using systematically the multi-body elements of Modelica Standard Library (Fig. 2). Each link is mathematically described as a rigid body, while relative rotations in space between links are allowed by so called revolute joints, which are then connected to 1-D transmission models. A graphical shape is associated to each body to provide a 3D virtual animation of robot.

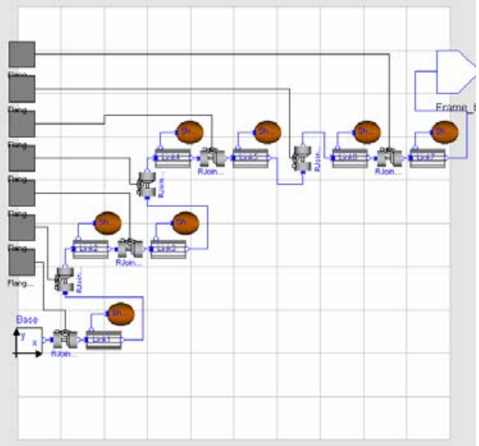


Fig. 2. SPIDER multi-body chain

B. Two-phase Brushless Motor

The SPIDER arm is equipped with two-phase brushless motors. The Modelica model has been built as an extension of the brushed DC motor available in the Modelica library. The schematic of the model is shown in Fig. 3. The customized block EMF_2 accounts for the electro-mechanical conversion and computes the back electromotive forces (BEMF) in the two quadrature stator winding. Harmonics in the bemf profiles cause harmonics (ripple) in the motor torque, whose frequency is proportional to rotor speed [14]. Besides the electromechanical conversion, the model describes the electrical dynamics of the two stator windings, the equivalent inertia of the rotor and the viscous friction of the motor. The model also includes a virtual encoder. The icon shows that the model interfaces are two electrical terminals and a mechanical flange [8], [9], as in a real motor.

C. Electronic Drivers

The NRPI module (Fig. 4) reproduces the dynamics of the analog motor driver used in SPIDER servos; it performs current regulation with sinusoidal reference (actually the frequency is proportional to rotor speed) on the motor phases, by means of two independent current loops (PI analog regulator, providing saturation and anti-windup compensation). To keep computational burden and time within reasonable limits, a linear amplifier is used in place of the PWM (pulse width modulation) amplifier.

The model is also able to reproduce torque ripple due to biases of the current sensor outputs.

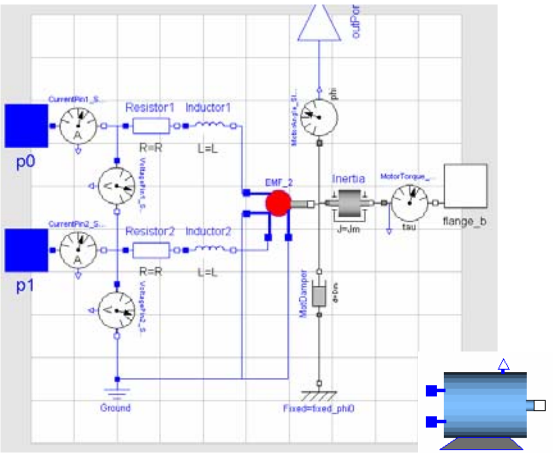


Fig. 3. 2-phase brushless motor schematic and icon

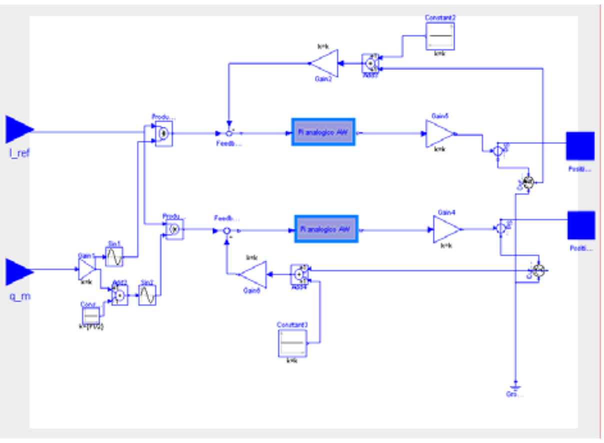


Fig. 4. Stator winding current control module

D. Gearbox

Gearboxes are the most important sources of friction, torsional flexibility, and backlash in the joints. Since these phenomena are responsible for vibrations and motion irregularities and limit the achievable bandwidths of position control loops, they must be modeled carefully. The gearbox model shown in Fig. 5 is obtained by aggregation of basic models available in the Modelica library and of a model developed on purpose, which implements the LuGre (Lund-Grenoble) friction model. The LuGre friction model equations [15] are:

$$\begin{cases} \dfrac{dz}{dt} = \dot{q} - \sigma_0 \dfrac{|\dot{q}|}{g(\dot{q})} z \\ g(\dot{q}) = \tau_c + (\tau_s - \tau_c) e^{-|\dot{q}/v_S|^{\delta_S}} \\ \tau_a = \sigma_0 z + \sigma_1 \dfrac{dz}{dt} + D\dot{q} \end{cases}$$

where τ_s is the stiction torque, τ_c is the Coulomb torque, σ_0, σ_1, and δ_S are two numerical parameters, v_S is the so called Stribeck velocity, D is the viscous factor, q is the rotational angle and z is the state variable of the model.

Thanks to the declarative and acausal nature of Modelica language, the implementation of these equations in a specialized 1D-rotational object (i.e., having two 1D rotational flanges as interfaces, each one characterized by the angular position φ and torque τ) is straightforward, requiring only the transcription of the equations written above.

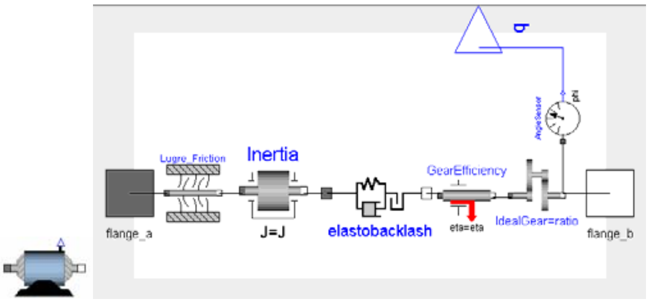


Fig. 5. Gear train schematic and icon

Parameter σ_0 above is a critical one: the higher it is the more “discontinuous” the friction characteristic is, but at the cost of heavier computational load.

E. Control Algorithms

Two control structures have been implemented and tested: the independent joint control, with P/PI cascaded regulators, closed on motor position feedback (the PI, inner loop) and on load position feedback [16], and the operational space hybrid control [17]. As far as control system coding is concerned, Modelica is endowed with the features common to most simulation environments.

F. Contact Model

Modeling of interaction with environment required the development of new objects, starting from the Modelica general multi-body flange (interface). In fact, the multi-body chain, defined for free motion conditions, can be used to simulate contact/proximity operations with the addition of proper interaction models. Two models have been developed: the elasto-viscous model, describing the environment as an elastic flat surface with a second order dynamic behavior, and the rigid model, assuming a perfectly rigid flat surface.

The rigid contact constraint is modeled by an object which extends a basic class of the Modelica library featuring just a pair of 6-D mechanical flanges. The extension consists in describing the constraint in terms of free / constrained motion directions of one flange with respect to the other. This is rather straightforward to do. Then, constraining the end effector requires just to connect (drawing a line) in a rigid way one flange of the new object to the end effector, and the other to the ground.

The computation burden for the numerical solution of the model with rigid contact is heavier than with elastic contact, nevertheless it gives more accurate results whenever the interacting environment is very hard.

Fig. 6 shows the arm mechanical chain model connected to either the interaction model or the payload. Also the connection to payload is straightforward thanks to symbolic equation processing of DYMOLA. The payload can be modeled as an independent body, standing still or moving freely, which gets rigidly connected to the end effector once a logic input states that it has been latched. Besides being advantageous from the modeling point of view, this features also increases the operational flexibility of the simulation model, since complex sequences of operations, made up of free space, contact, with and without payload operations can easily be simulated.

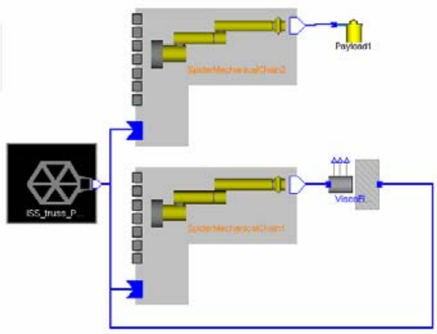


Fig. 6. Mechanical chain with payload and interaction model

IV. VALIDATION OF THE JOINT MODEL

The joint model, made up of motor, driver, and gearbox modules, has been validated thanks to data available from an experiment on a single SPIDER joint performed onboard EUROMIR [18], and from laboratory experiments on the same joint [19]. The aim of experiments is finding out the nature and level of disturbances generated by the joint and their sources. Torque ripple harmonics estimated from the onboard experiment are shown in Figure 7. The motor was turning at the nominal speed of 955 rpm, and the three main harmonics are at 111.4Hz (p times the motor frequency ω, being $p=7$ the number of pole pairs), due to current sensor biases, at 222.8Hz ($2p\omega$), due to cogging and phase torque unbalances and at 445.6Hz ($4p\omega$), due to BEMF shape imperfections. The model is able to predict amplitude and frequency of these harmonics by setting a value of 1.5% of current offset, 1.5% of phase torques unbalance, and 0.8% for the relative weight of the third BEMF harmonic. These values are in good agreement with the experimental data measured on a motor prototype at the motor manufacturer laboratories.

V. SIMULATION RESULTS

The SPIDER model and related simulation and interface environments are profitable for testing different control architectures and for simulating various operating conditions which may be encountered by a robotic system during a space mission. Some illustrative examples follow.

A. Microgravity Disturbances Assessment

One of the most important issues involved in the certification of space systems to be installed on the International Space Station concerns their compliance to microgravity disturbances requirements established by NASA [20]. Such compliance can be assessed for the SPIDER arm by simulation of a careful model, not only able to reproduce reliably the gross motion dynamics, but also the torque disturbances and the resonant behaviors due to joint flexibility. Fig. 8 shows the model prediction of the disturbances transmitted to ISS main truss by SPIDER manipulator with a 30Kg payload and moving at 2.5 cm/s. From this prediction it turns out that the SPIDER arm should be able to comply to NASA requirements.

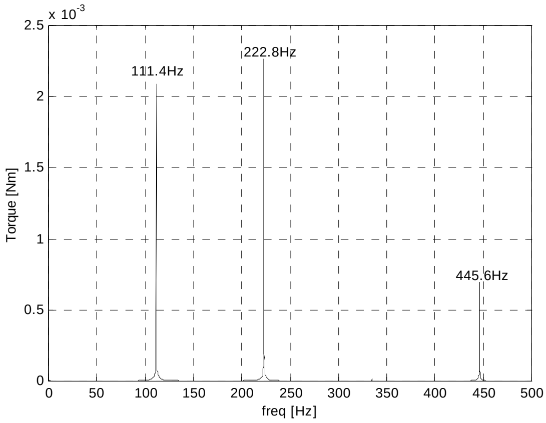


Fig. 7. Torque ripple harmonics estimated from experiments

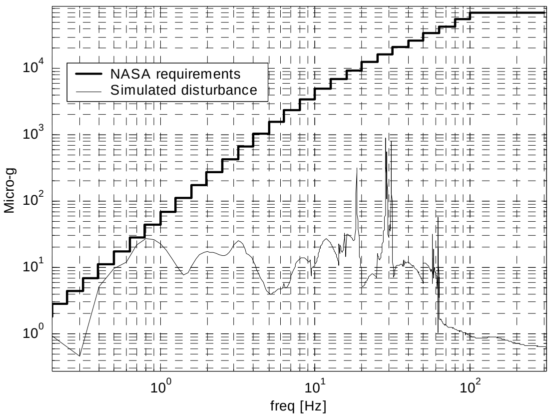


Fig. 8. Microgravity disturbances: SPIDER arm plus 30Kg payload

B. Safety Requirements under Faulty Condition

Other project constraints for a space robotic system concern safety requirements. For instance, it must be verified that in case of accidental impact against an obstacle (ISS) at maximum operating speed with simultaneous complete fault of control system determining maximum current in the motors, the impact energy and force do not exceed critical thresholds. The results of the simulation of such an extreme condition are given in Fig. 9, where Vci is the power supply voltage and K is the contact elasticity of the obstacle.

C. Force control with Rigid Contact

Plots shown in Fig. 10 were obtained during simulations of the SPIDER model aimed at tuning and assessing performances of hybrid position / force control algorithms. Setpoint and actual contact force are plotted for contact operations against a perfectly stiff environment, modeled as a kinematic constraint.

VI. REMOTE USE OF THE MODEL THROUGH THE INTERNET

A client-server interface, called SIMECS-Remote Interface [21] allows an authorized user to simulate the SPIDER model from a remote station through the Internet. More details on the model and on the remote interface can be found at the web page www.elet.polimi.it/res/simecs/. The SIMECS-RI application is a visual simulation environment dedicated

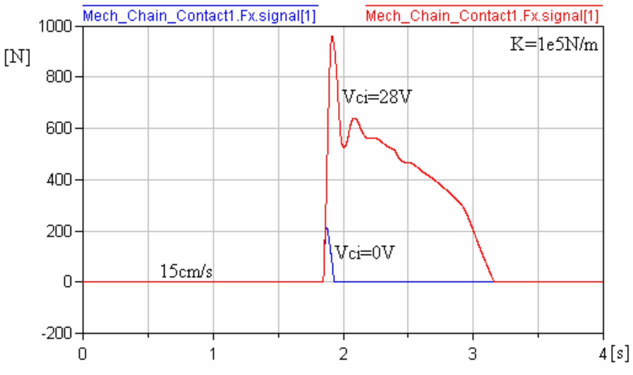


Fig. 9. Simulation of impact with ostacle and fault of control system

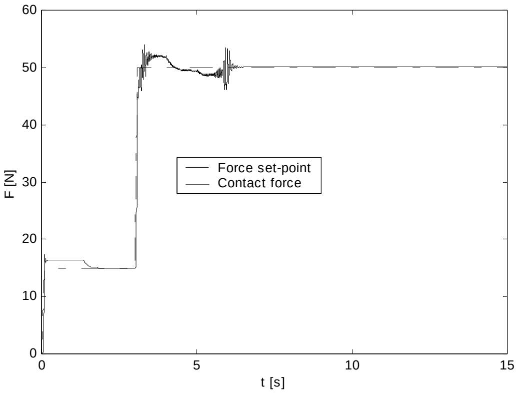


Fig. 10. Contact force with a perfectly stiff surface

to simulations of executions of commands assigned to robotic arms which operate in space. Simulations run on a server computer, and are performed by an application which is external to the SIMECS-RI server, while simulation results are presented on a client computer. The application is conceived to be flexible enough to deal with any kind of manipulator. This means that it should be able to present information about any kind of mechanical chain and motion control systems, and it should be also able to initialize and run simulations of robots which accept commands at different abstraction levels. Information to be presented about the robot model consists of robot model documentation, model parameters and main variables, and 3D graphic and kinematic models of the robot.

The user is allowed to choose a robot model from a library of available models, to assign model parameters and initial state, to define, assign, and simulate the execution of a robot command, and finally to analyze the simulation results. Depending on the model inputs, low level setpoints (e.g., motor current setpoints), joint and Cartesian space paths, and high level commands can be assigned to the robot model. Fig. 11 shows a snap-shot of the remote interface.

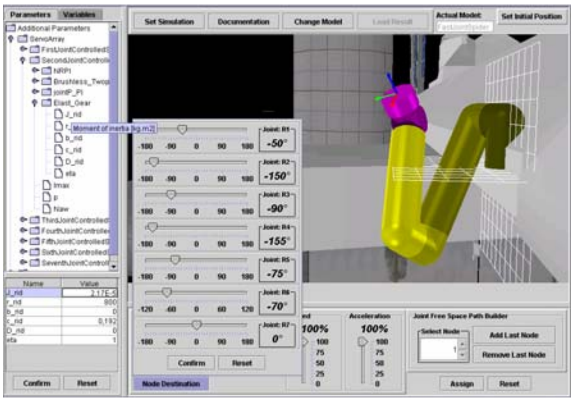
VII. CONCLUSIONS

The simulation model of a 7-DOF space arm has been obtained, exploiting the powerful features of the object-oriented, physical modeling language Modelica. The mechanical chain is modeled as a rigid multi-body chain. Special care is given to modeling of brushless motors and of their torque disturbances, and to friction and torsional flexibility in the joint gear trains, which affect motion quality to a large extent. Contact between robot and environment in both rigid and elastic conditions are considered. Models of motors, gear trains and control electronics have quite natural interfaces (electrical terminals and mechanical flanges), which facilitate the assembly of the overall model, enhancing readability of the overall model and reuse of component models.

The model of joints has been tuned and validated based on data gathered in an experiment onboard the MIR station. Simulation of the model have confirmed that the arm and its controller are compliant to NASA's microgravity disturbance requirements and can tolerate some critical faulty conditions.

ACKNOWLEDGMENTS

This work has been supported by ASI (Italian Space Agency) under contract no: I/R/217/02.



Name	Value
J_rid	2.17E-5
r_rid	800
b_rid	0
c_rid	0,192
D_rid	0
eta	1

Joint: R1 -50°
Joint: R2 -150°
Joint: R3 -90°
Joint: R4 -155°
Joint: R5 -75°
Joint: R6 -70°
Joint: R7 0°

Acceleration 100%

Select Node 1

Fig. 11. The SIMECS-RI user interface

REFERENCES

[1] J.C. Piedboeuf, M. Doyon, P.Langlois, R. L'Archeveque: *SYMOFROS: A Flexible Dynamics Modeling Software*, 5th International Symposium on Artificial Intelligence, Robotics and Automation in Space, ESTEC, June 1999, ESA SP-440, pp. 709-716

[2] DCAP (Dynamic and Control Analysis Package), http://esapub.esrin.esa.it/bulletin/bullet87/franco87.htm

[3] X. Cyril: *ROSESAT – A Graphical Spacecraft Simulator for Rapid Prototyping*, Space OPS98 Symposium, June 1998, Tokyo, Japan, Paper ID 4c002

[4] J.P.L: DARTS home page, http://dshell.jpl.nasa.gov/dshell.html

[5] MSC.ADAMS home page, http://www.mscsoftware.com

[6] P. Masarati, M. Morandini, G. Quaranta, and P. Mantegazza, Open Source Multibody Analysis Software, Multibody Dynamics, IDMEC/IST, Lisbon, Portugal, July 2003

[7] G. Ferretti, G. Magnani and P.Rocco, "Virtual prototyping of mechatronic systems in Modelica, " Proc. IFAC Conf. on Mechatronic Syst., 2002, pp.865-870.

[8] http://www.Modelica.org/ : *Modelica – A Unified Object-Oriented Language for Physical Systems Modeling - Tutorial* , 15-12-2000

[9] http://www.Modelica.org/ : *Modelica – A Unified Object-Oriented Language for Physical Systems Modeling – Language Specification* , 10-7-2002

[10] S.E. Mattsson, H. Elmqvist and M. Otter, "Physical system modeling with Modelica, " Contr. Eng. Pract., **6**, pp.501-510.

[11] K.E. Brenan, S.L. Campbell and L.R. Petzold, "Numerical solution of initial-value problems in differential algebraic equations," North Holland, 1989.

[12] C.C. Pantelides, "The consistent initialization of differential-algebraic systems," Siam J. of Sc. and Stat. Comp., **9**, 1988, pp.213-231.

[13] R. Mugnuolo, S. Di Pippo, P.G. Magnani and E. Re, "The SPIDER manipulation system (SMS) – The Italian approach to space automation," Rob. and Auton. Syst., **23**, 1998, pp.79-88.

[14] G. Ferretti, G. Magnani, P. Rocco, "Modelling Identification and Compensation of Pulsating Torque in Permanent Magnet AC Motors", *IEEE Trans. on Industrial Electronics*, **45**, 1998, pp. 912-920.

[15] C. Canudas de Wit, H. Olsson, K. J. Astrom, P. Lishinsky, "A New Model for Control of Systems with Fricrion,", *IEEE Trans. on Automatic Control*, **40**, 1995, 419-425.

[16] G. Ferretti, G. Magnani, P. Rocco, "Modelling and Control of Servomechanisms", RAMSETE, Lecture Notes In Control and Information Sciences, Springer, 2001

[17] O. Khatib, "A unified approach for motion and force control of robot manipulators: the operational space formulation, " IEEE J. of Rob. and Autom., **3**, 1987, pp.43-53.

[18] Tecnospazio S.p.a, *EUROMIR '95 Experiment, Final Presentation*, Roma, 5/9/96

[19] Etel S.p.a, *Motor / Driver for EUROMIR, Tests Report*, Doc. TSEU201, 4/1/95

[20] NASA, SSP 57003 Revision A, *Attached Payload Interface Requirements Document – International Space Station Program*, 5/8/2002, pp.16-24

[21] G. Ferretti, M. Gritti, G. Magnani and P.Rocco, "A Remote User Interface to Modelica Robot Models", Proc. Modelica Conference, 2003, http://www.modelica.org/Conference2003/papers.shtml