THE NTUA SIMULATORS FOR SPACE ROBOTS ON ORBIT

Georgios Rekleitis[(1)], Ioannis Tortopidis, Iosif Paraskevas, Dimitrios Psarros, Ioannis Kaliakatsos, Ioannis Roditis, and Evangelos Papadopoulos[(2)]

Department of Mechanical Engineering, National Technical University of Athens, Athens 15780, Greece
[(1)] georek@central.ntua.gr, [(2)]egpapado @central.ntua.gr

ABSTRACT

The importance of simulators as testbeds for proposed space robotic systems is unquestionable in our days. In this paper, the NTUA approach on both software and hardware space robots simulators, is presented. The software simulator is fully parameterized in order to be capable to simulate any system that consists of a base and a number of serial appendages. The hardware simulator emulates 2D motion in zero gravity, and consists of a two-manipulator space robot moving on top of a granite table. The NTUA simulators are currently at the final stages of their development.

1. INTRODUCTION

The commercialization of space introduces the need for robotic devices that can assist humans in the construction, repair, protection and maintenance of space stations or satellites. Such robotic systems are very expensive to build and even more to put successfully on orbit. Furthermore, any error, besides the possible loss of an expensive system, can threaten humans, such as astronauts that may be working with the robot. It is obvious that such systems must be designed and tested to be foolproof before they are tried in their remote, operating environment. Since real experimental tests are practically out of the question, a simulator is the only remaining solution.

Simulators in general can be of two kinds. These are software and hardware simulators. Software simulators have the benefit of allowing us to do almost anything we want. The problem is that reality has so many unknown and unmodeled factors, that at best we can approximate reality. Nevertheless, an explicit model of the dynamics of the system can result in a simulation very close to reality. For that reason, several software simulators have been developed over the years, in order to test proposed systems and motion strategies, as in [1].

The main problem encountered while designing a hardware space simulator, is the existence of gravity. Several methods to simulate zero gravity have been proposed. One such approach uses water tanks and systems neutrally buoyant, [2]. The advantage of this approach is that it simulates space motion in 3D, but the existence of the water resistance hampers the realism of the simulation, thus making this approach better for training astronauts in zero gravity slow motions. Another approach is the use of a mechanism that supports the tested robotic system, negating the gravitational force, [3], [4]. This approach, although promising, has yet to overcome the problems due to manipulator singularities, that do not allow any desired motion. Other approaches include throwing systems in deep wells, or parabolic flights, but those are severely limited by the time available for experiments. Yet another approach is a planar simulator, based on practically frictionless motion of the simulated robotic system on a horizontal plane. This motion can be achieved by several methods, such as the use of air bearings [5, 6]. This can be as realistic as close to the actual frictionless motion we can get, but has the obvious disadvantage of the 2D motion restriction.

In this paper, the simulators developed at the National Technical University of Athens (NTUA), are presented. These include a software simulator with an animated graphic representation and a planar hardware emulator, based on the frictionless motion of a robot on a granite table, by means of air-bearings. These systems are described in some detail next.

2. SOFTWARE SIMULATOR

The software simulator for space robots on orbit consists of three basic components. These are (a) the dynamic equations of motion, (b) the numerical simulation including various control algorithms and, (c) the animated graphic representation. The equations of motion are obtained using the mathematical package Mathematica®, while the system's behavior under the chosen control method is simulated using a simulation package, such as Simulink®. The animated graphic representation is realized in a program developed in our lab at the NTUA, and is based on OpenGL libraries. This program uses the Simulink® model data output, and produces the desired animation.

2.1. Dynamic Analysis

An orbital robotic system consists of a base with several appendages, such as manipulators, communication antennae etc. The motion of any of these appendages changes the geometry of the system, thus influencing its dynamics. The system dynamics depend on the kinematics (structure, number of degrees of freedom (dofs)), and on parameter values. Such a system in 3D has at least six dofs for the base position and orientation, plus as many as the number of the joint variables of the manipulators. Thus, the user of a simulator for orbital space robots must be able to define, not only the inertial and geometric properties of the base, but also the number and the location of the manipulator base, as well as the inertial and geometric properties of their links.

To begin the dynamic modeling of such a system, a set of generalized coordinates q_i is chosen, describing the position and orientation of the base, as well as the position of the links of the manipulators. In order to obtain the dynamic equations of motion, the Lagrange formulation is used. The Lagrange function or *Lagrangian* of the system is defined as

$$\mathcal{L} = T - U \tag{1}$$

In Eq. (1), T and U are the system's total kinetic and potential energy respectively. Note that the systems' total potential energy is equal to zero for orbital systems. Thus, Lagrange's equations of motion for each generalized coordinate are obtained based on the kinetic energy of the system,

$$\frac{d}{dt}\frac{\partial T}{\partial \dot{\mathbf{q}}} - \frac{\partial T}{\partial \mathbf{q}} = \mathbf{Q} \tag{2}$$

The final form of the equations of motion, is obtained as

$$\mathbf{H}\ddot{\mathbf{q}} + \mathbf{C} = \mathbf{Q} \tag{3}$$

In Eq. (3), $\mathbf{q}$ is the k x 1 vector of the generalized coordinates, $\mathbf{H}$ is a k x k matrix related to the inertia properties of the system, $\mathbf{C}$ is a k x 1 vector containing all the nonlinear velocity terms and $\mathbf{Q}$ is the k x 1 vector of the generalized forces.

Matrices **H** and **C** are obtained symbolically using Mathematica®. The code that generates these matrices can be easily modified to account for the number of manipulators and their links that the simulated robotic system has. The inertia and geometric properties of the system are introduced as code parameters. As stated above, in order to acquire simulation data to use in the animation code, the obtained matrices need to be converted into Matlab/ Simulink®. Thus, the inertial and geometric properties of the system remain as parameters that can be changed in Simulink®, providing easy data acquisition for different simulated systems, as long as the number of manipulators and links, as well as the general shape of the base (e.g. cube, cylinder) remain the same. If any of these parameters (e.g. the number of manipulators) is changed, new **H** and **C** matrices are obtained. The new dynamic model can then be imported in Simulink®, and joined with the desired control method. This control method depends strongly on the desired response of the robotic system as well as on its properties.

2.2. Graphic Simulator

In order to visualize the results of the simulations, an animation program was developed. Using this program the user can have a better view of the behavior of the systems that have been modeled, similar to the behavior in the real world.

The program is created to run under Windows. It is developed using the C programming language and the Application Programming Interface (API) OpenGL for the creation of the 3D environment. To make the code easier, the *glut* functions library was also utilized. The program has two parts, one for creating a space robot, and one for moving it according to the data created by the simulation.

The idea is to create each robot with the minimum possible number of parameters. The program uses these parameters to draw the robot using simple shapes and surfaces. These parameters concern the body of the robot, the manipulators, the thrusters used for moving in space and the antenna(e) used for communicating with earth and other space robots.

The space robot base can be a rectangle, a cylinder or an orthogonal regular prism. Additional data for the dimensions of the base are asked by the program. Then the number of manipulators, and the position and orientation of their base coordinate systems, must be provided to the program. The manipulators are drawn using the Denavit - Hartenberg parameters. Then, the main body (base) thrusters are placed by entering their location and orientation. The antennae are similarly drawn. Figure 1 shows a space robot with a cylindrical base, twelve thrusters and two manipulators with four joints each, created using the above steps. Note that each robot is created once and then can be used in different simulations.

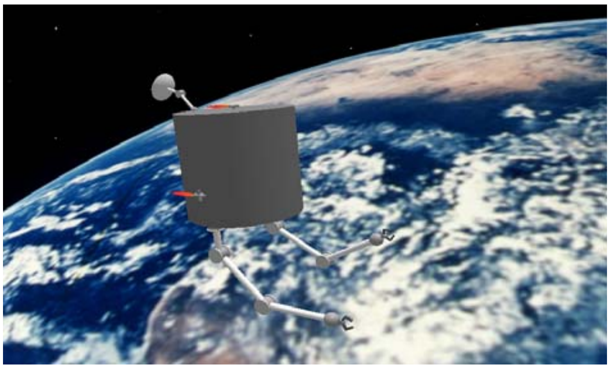

Figure 1. The space robot moving on orbit.

The data for the robot motion, as already mentioned, comes from the model running in Simulink®. The time interval between two frames in the animation program is 50 ms, so a fixed step of 50ms is used in the simulation. The total time that the animation can last is 500 s (10,000 frames). We use a Matlab® function to pass the simulation data to files, in a format that the animation program can read. The files are copied by the user to the folder that contains the saved model of the graphic representation of the robot. Then, the animated graphic representation of the simulation can start.

We can follow the motion of the robot using a fixed camera or one that moves with it. Other options offered by the program include the ability to draw the path of the robot, the provision of three additional cameras that display the path of the robot, a coordinate system that displays the direction of the robot and one that displays the direction of the path of the robot.

3. HARDWARE SIMULATOR

Besides the software simulator, an experimental apparatus is being developed at the NTUA, in order to be used as a testbed for motion in zero gravity environments. After considering several ideas for testbeds, a planar simulator on a flat surface was chosen, as the best in terms of both reliability of simulations and construction feasibility. The concept behind this idea is that a robotic mechanism will be able to perform frictionless motion on a flat horizontal surface, thus simulating zero gravity planar motion, since the only external forces acting on the robot, that is gravity and the force acting from the table, will cancel out each other. The problem is how to achieve frictionless motion on a horizontal table. This is tackled by using air bearings. These are small porous graphite bearings, on top of which the entire robotic mechanism is mounted and to which gas under pressure is supplied. The gas is used to create a very thin film under the bearing (usually of several microns), on which the whole system floats.

3.1. Design of the simulator

First and foremost, the "flat horizontal surface" on which the robotic mechanism would float, had to be chosen. In order to be able to float on such a thin film of gas, the robotic mechanism would have to move on a very flat surface. The flatness would have to be both microscopic, since the local roughness of the surface should have to be significantly less than the width of the gas film, and macroscopic, in order to avoid a wavy surface, that could render the robotic mechanism unstable. A granite surface was chosen, with special finishing to achieve the desired local and macroscopic roughness. This surface has a maximum anomaly height equal to 5 μm. Supporting legs, whose length can be adjusted, ensured that the top granite surface is perpendicular to the gravitational force, thus ensuring the desired cancellation between the two external forces acting on the robotic mechanism.

The robotic system moving on the granite table is a small platform (base) carrying the air bearings, a set of jets and a reaction wheel for main base actuation, two manipulators for interaction with other objects and other supportive equipment, see Figure 2.

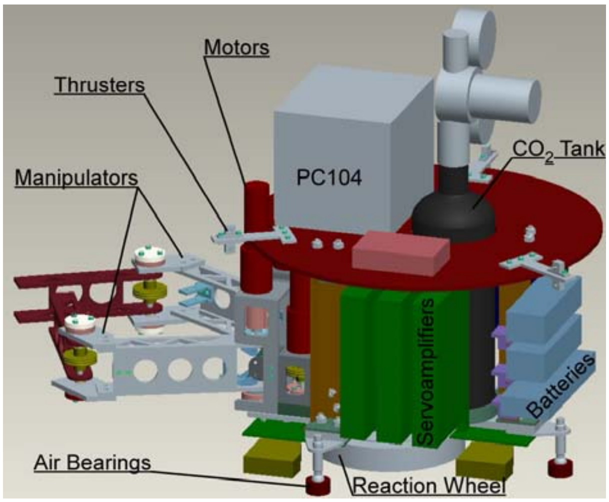

Figure 2. A 3D CAD model of the space robot.

The requirements for the robotic system are the following:

- Light weight and small size
- Maximum autonomy
- Capability for interaction with the environment

Light weight and small size were major considerations during the design of the robotic mechanism. Compact solutions for the various issues of the mechanism, especially on the transmission system for the motion of the manipulators, were derived, in order to obtain a compact and lightweight robot.

Autonomy is achieved in three ways: propulsion autonomy, computational power autonomy and electrical autonomy. A tank that is mounted on the base of the robot, containing CO_2 under 60 bar, provides CO_2 both the air bearings and the jets. Therefore, no external hoses are used. A set of regulators and filters ensures the smooth operation of both subsystems.

Computational autonomy is achieved by the use of a PC104 tower. This way, the robot does not have to communicate with an external computer in order to plan a controlled motion, based on the feedback it receives from its sensors.

Electrical autonomy is achieved by the use of batteries that provide the needed power for the reaction wheel, the motors, as well as the PC 104 unit and any other electrical circuit required for the smooth operation of the mechanisms' several subsystems.

The interaction with the environment is achieved via a pair of manipulators. In order to do so, the robotic mechanism must be able to know its position and orientation on the granite table, at each moment of the simulation. This is achieved using position sensors. A number of proposed types of sensors are discussed in Section 3.3. The interaction via the manipulators is done in such a way that they can capture a floating object that acts as an orbital debris without actuation. The two manipulators have two links each. The second link of each manipulator is driven through a system of pulleys. The design of each manipulator and the corresponding set of pulleys is done in such a way, that the motion of the first link will not affect the motion of the second one. The manipulators are displayed in Figure 2.

3.2 Base actuation

The purpose of the actuation system is to provide the robot with the necessary forces and torques for its various movements. It consists of two main systems, namely the thrusters system and the reaction wheel.

3.2.1 *Thruster system*

The thruster system comprises of two main subsystems: the fluid monopropellant subsystem, and the electrical/ electronic subsystem. The fluid monopropellant subsystem consists of a storage tank capable of enclosing 540 g of CO_2, having an initial pressure of 60 bar, two pressure regulators, one for holding the pressure to the 7 bar level for propulsion purposes, and the other for throttling the pressure from 7 to 4.1 bar for use by the air bearings, the thrust nozzles (thrusters), and the various piping connecting these components.

The electrical/ electronic subsystem consists of the solenoid valves that control the flow of the CO_2 in the thrusters, and the Pulse Width Modulation (PWM) actuation circuit for driving the solenoid valves. This circuit receives a logical signal $0-5V$ in PWM format, and outputs a PWM signal, on a higher voltage scale than the input one ($0-24V$).

One of the problems we had to address was to determine the value of thrust we can obtain from each thruster, when CO_2 flows through it. To do so, we used a force sensor and actuated the solenoid valves with a 7Hz PWM signal, while varying the duty cycle from 0% to 100%. As can be seen in Figure 3, apart from a linear region (C), which we expected, there is also a dead band (A) and a saturation region (D), which are related to the valve's time constant (signal too fast to activate the valve). Regions B are essentially settlement regions from the linear one C, to the nonlinear A and D.

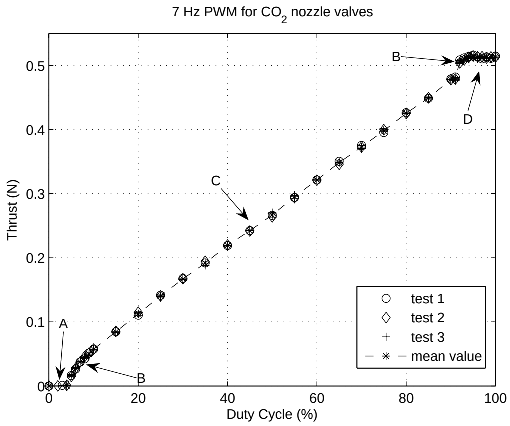

Figure 3. Thruster force vs. valve PWM duty cycle.

3.2.2 *Reaction wheel*

In an effort to reduce propulsion gas consumption, we are developing a reaction wheel, i.e. a momentum exchange attitude control actuator, providing torque around an axis vertical to the plane of motion. Its operation is based upon the conservation of momentum principle, as illustrated in Figure 4.

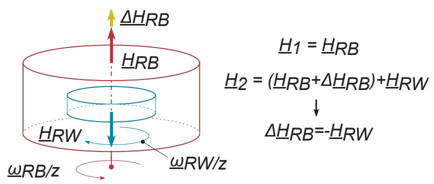

Figure 4. Reaction wheel principle of operation

For the purposes of the space simulator, it is preferred to use reaction wheel instead of other attitude control actuators due to the simple control algorithm, attitude fine tuning and the potentially best fit to the two-dimension experiment.

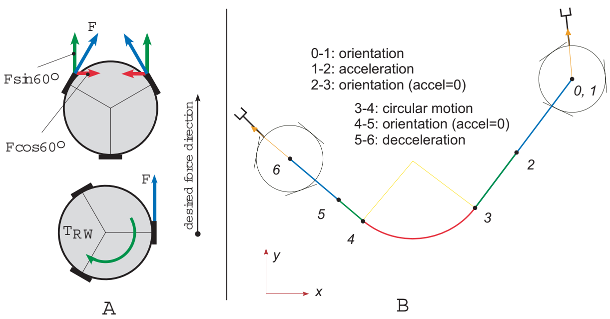

Figure 5. A. Force generation, B. Reference trajectory.

As a first step of the design approach, general limitations about force generation were determined. With a reaction wheel installed, it is convenient to generate a force using one thruster parallel to the desired force direction, while the reaction wheel counteracts unwanted torques, as shown in Figure 5A. A curve consisting of two straight lines and a circular sector, as the one shown in Figure 5B, was adopted as a reference trajectory used in comparing the two options of motion. Total thrust given by Eq (4), was considered as an appropriate criterion in evaluating the two cases.

$$Thrust = \sum_i \int_t f_i(t)\,dt \qquad (4)$$

A significant reduction of reaction wheel performance may occur due to motor saturation. A Simulink® model for simulating four quadrant motor operations was developed, taking into consideration operating range and temperature raise during operation. If the required torque exceeds motor maximum output (for instant angular speed), three appropriate thrusters have to be activated to produce the torque difference. After simulating various initial and final configurations, it was found that by using a reaction wheel, propulsion gas consumption decreases at a percentage of about 50%. Figure 6 presents each thruster force applied versus time for thruster-only motion (left) and thruster plus reaction wheel combined motion (right).

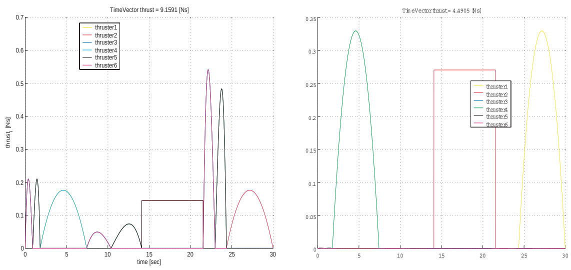

Figure 6. Thrust comparisons.

A single dof manipulator was added to the model, to simulate the task of debris capturing, and even with the manipulator extended, combined base-wheel motion is proven to be superior to the thruster-only case. A variety of flywheels were tested to identify the relationship between wheel moment of inertia and gas consumption.

After debris collection, force and torque requirements are highly increased. Depending on robot initial/ final configuration, motion duration and debris mass, it is possible to confront lower consumption at the thruster-only motion. However, the sum of the total thrust including motion towards debris and following its capture, is lower at combined jets and reaction wheel motion. To reduce use of thrusters at the second part of motion, a trajectory of similar shape is selected, guiding the center of gravity of the whole system (robot, reaction wheel, manipulator, debris). This leads to improved moment of inertia distribution. Simulation showed even lower consumption for the combined motion, after the last consideration.

Consumption decrease rate may vary depending on the motor, flywheel mass properties and motion characteristics. Motor saturation must be avoided as it results to thruster activation. This additional actuator, not only can provide portion of the required torque, but also allows us to extend our range of experiments.

3.3 Joint actuation and electrical autonomy

In order to determine manipulator gearmotors, two sets of trapezoidal profile trajectories were simulated, both using an estimation of system parameters. The first set of trajectories included movements in which the acceleration and deceleration phases were of an equal time duration and magnitude. The second set of trajectories was defined assuming a desired maximum attained velocity, with no constant velocity segment, i.e. triangular profile. The results of the simulations, gave an estimate for torque, T (max, rms) [Nm], and power, P (max, rms) [W], for each joint, providing thus a guide in selecting the gearmotors.

The electrical supply consists of Li-Polymer batteries, whose mass and volume is less than that for NiMH or Lead-Acid types. In addition, Li-Polymer batteries can provide high current peaks, necessary for high acceleration – deceleration phases. However, during the coasting phase, the current rms value remains relatively small. Two levels of voltages are used. The first one, of about 7.5 V, provides power to the electronics such as the PC104, and the logic circuits. The second source, of about 24 V, provides power mainly to motors and solenoid valves.

3.4 Sensing

In order to acquire high precision position and velocity data for the objects on the table, a number of proposed concepts were considered.

A first choice includes the use of cameras, the actual number and position of which depends on their technical specifications. This solution has been used before. An appropriate analysis of the output provides the location and orientation of a large number of objects potentially moving on the granite table, without any need to add special electronic or mechanical components on them. On the other hand the cost for cameras with adequate resolution is high.

Ultrasonic devices are a second option. Each object on the table can have a number of ultrasonic transmitters on known positions, depending on the object's shape and the necessary precision, while around the table ultrasonic receivers are placed. At specific time intervals, an IR device sends a signal in order to initialize the timers at all devices, while at the same moment an ultrasonic pulse will be transmitted. Considering the time of flight of this pulse to several receivers, an algorithm much like the ones used by GPS, can compute the position and velocity of the object. The cost of this design is relatively low, with no need for pattern recognition, however the signal noise and the construction complexity are rather difficult problems.

Another possible choice is the utilization of an Inertial Navigation System (INS), such as a set of accelerometers and gyroscopes. The disadvantage this method has is the inherent accumulation of odometry errors, which is by no means negligible. Commercial INS units with very small odometry error accumulation, are considered to be a rather expensive solution.

Finally the case of optical sensors provides another option. The system uses mouse technology with very high resolution (2000 dpi). The technology employs LED or laser emitters to light the surface on which the sensor is moving, collecting pictures at a high rate. By appropriate manipulation of the received pictures, the position of the mouse cursor is extracted. Thus, each sensor provides the X-Y position of a certain point of the robot. It is clear that, at least two such sensors are needed in order to know at each moment the position of the robot. Nevertheless, a third sensor can be utilized in order to correct any accumulated odometry errors, which are by far smaller that the ones of an INS system.

The kind of sensors to be used, will be decided after a thoughtful investigation of a number of parameters, considering system complexity, algorithm complexity, precision, ease of use, and cost. However, a mixed solution, assisted by sensor fusion, is also a possibility.

3.5 Planning and control

The simulator is an excellent testbed to study the validity of proposed trajectory planning and control methods. Among other proposed methodologies, the simulator will be the ideal platform to experimentally analyze the concept of capturing tumbling objects using the percussion point of the robot's links, [7]. According to this method, a reduction of the forces affecting the base of a space robot can be accomplished, leading to less fuel consumption and higher system endurance.

The case of a free-flying base can also be examined. The ability to move the robotic system through desired trajectories, without using any of the jets is of high importance, and this platform gives the opportunity to test these highly sophisticated control algorithms, [8].

The number of control experiments that can be accomplished on this testbed is large. The cooperation of many robots, the handling of objects and debris collection are some of these.

4. CONCLUSIONS

In this paper, the work done at the NTUA on the development of both software and hardware space simulators is presented. The graphic simulator is performing smoothly and already simulates a number of moving space robots. The hardware emulator is on the final stages of construction and our intermediate experiments promise a successful operation.

ACKNOWLEDGEMENTS

Support of this work by the EPAN Cooperation Program 4.3.6.1.b (Greece-USA 035) of the Hellenic General Secretariat for Research and Technology is acknowledged.

REFERENCES

[1] B. Schafer, R. Krenn, B. Rebele. "On inverse kinematics of redundant space manipulator simulation", *Journal of Computational and Applied Mechanics*, Vol. 4, No. 1, pp. 53-70. 2003.

[2] http://www1.jsc.nasa.gov/dx12/site/index.shtml.

[3] W.-H. Zhu, J.-C. Pieboeuf, Y. Gonthier. "Emulation of a space robot using a hydraulic manipulator on ground", *Proc. 2002 IEEE Int. Conference on Robotics & Automation*, pp 2315-2320, Washington DC, May 2002.

[4] West, H., Papadopoulos, E., Dubowsky, S., and Cheah H., "A Method for Estimating the Mass Properties of a Manipulator by Measuring the Reaction Moments at its Base," *Proc. of the IEEE Int. Conference on Robotics and Automation*, Scottsdale, AZ, May 1989, pp. 1510-1701.

[5] K. R. Zimmerman, R. H. Cannon, Jr. "Differential carrier phase GPS techniques for space vehicles randesvous", *Proc. Inst. Navigation GPS-94 Conf.*, Salt Lake City UT, Sept. 1994.

[6] S.-J. Chung, E. M. Kong, D. W. Miller, "SPHERES Tethered formation flight testbed: Application to NASA's SPECS mission", *SPIE 5899-22, UV/Optical/IR Space Telescopes: Innovative Technologies and Concepts II Conference*, San Diego, CA, August 2005.

[7] E. Papadopoulos, I. Paraskevas, "Design and configuration control on space robots undergoing impacts". *Proc. 6th Int. ESA Conference on Guidance, Navigation and Control Systems.* 17 – 20 Oct., Loutraki, Greece, 2005.

[8] E. Papadopoulos, I. Tortopidis, K. Nanos, "Smooth planning for free-floating space robots using polynomials", *Proc. IEEE Int. Conference on Robotics and Automation,* pp. 4283-4288, Barcelona, Spain, April 2005.