MULTIBODY DYNAMICS 2005, ECCOMAS Thematic Conference
J.M. Goicolea, J. Cuadrado, J.C. García Orden (eds.)
Madrid, Spain, 21–24 June 2005

TRAJECTORY OPTIMIZATION AND REAL-TIME SIMULATION FOR ROBOTICS APPLICATIONS

Michele Attolico*, Pierangelo Masarati*, and Paolo Mantegazza*

*Dipartimento di Ingegneria Aerospaziale
Politecnico di Milano
Campus Bovisa, via La Masa 34, 20156 Milano, Italy
e-mail: michele.attolico@polimi.it, pierangelo.masarati@polimi.it,
paolo.mantegazza@polimi.it,
web page: http://www.aero.polimi.it/

Keywords: Trajectory Optimization, Real-Time Simulation, Space Robotics, Computational Mechanics, Open-Source Software.

Abstract. *This paper illustrates the use of the general-purpose multibody free software MBDyn for trajectory optimization in space robotics.*

Trajectory optimization requires the capability to explore the space of the feasible solutions which implies simultaneously satisfying the dynamics equations of the system and equality and inequality constraints.

A well-known and widely adopted technique is based on shooting, i.e. integrating in time the dynamics equations of the system, to compute the objective function and the constraints. When they are both differentiable, their derivatives with respect to the optimization variables can be numerically estimated by difference equations, and Sequential Quadratic Programming (SQP) techniques can be used to search the optimal set of variables.

Among the advantages of this technique, any valid means to generate the objective function can be used, including multibody analysis; this allows to introduce a high degree of flexibility and detail in the model of the problem.

A distinguishing feature of the presented approach is that a single general-purpose software is used for both the optimization and the real-time hardware-in-the-loop simulation of the problem, with a nearly unparalleled commonality of models and modeling environment.

1 INTRODUCTION

This paper illustrates the use of the general-purpose multibody free software MBDyn [4] for trajectory optimization in space robotics.

A distinguishing feature of the presented approach is that a single general-purpose software is used for both the optimization and the real-time hardware-in-the-loop simulation of the problem, with a nearly unparalleled commonality of models and modeling environment [1].

The optimal path planning of a multi-degree of freedom mechanical system is a typical optimization problem. The first step to solve this kind of problems requires to implement the equations that govern the dynamics of the system, and insert them into the optimization code. A multibody analysis code is a powerful tool to avoid this step, in fact it allows to model the system dynamics with a high level of detail and fidelity, based on the user's capability to prepare an appropriate model in terms of structural deformability, kinematic description of the mechanism, and more, without the need to write dedicated equations. Furthermore, using the multibody approach, the optimization algorithm becomes more versatile because many different problems can be solved by simply changing the multibody model.

In this work, the standard SQP optimization code Harwell VF13, with MBDyn as the multibody analysis tool, is applied to the optimal path planning of a 2 degree of freedom robotic arm. The goal is to find the behavior of the generalized torque that needs to be applied to the joints to produce an optimal path, based on an appropriate definition of the cost function. Usually, the optimization algorithm is iterative and needs an initial estimate of the unknowns to start. By using a simple multibody model, a first coarse solution can be obtained that constitutes the initial estimate of a new optimization process with a more sophisticated model.

The generality of the approach allows to use the same modeling environment for models and analysis of increasing sophistication, as well as for the final real-time simulation assessment of the resulting optimal solutions.

2 OPTIMIZATION PROBLEM

The minimization problem is formulated in terms of a cost function

$$f\left(\boldsymbol{u}\left(t\right),\boldsymbol{p},T\right)=\int_{0}^{T}\left(d_{1}+d_{2}\boldsymbol{v}\left(t\right)^{T}\boldsymbol{v}\left(t\right)\right)\,\mathrm{d}t, \tag{1}$$

where $\boldsymbol{v}\left(t\right)$ result from the control inputs $\boldsymbol{u}\left(t\right)$ after filtering by convoluting with a function $\boldsymbol{w}\left(t\right)$, i.e.

$$\boldsymbol{v}\left(t\right)=\int_{0}^{t}\boldsymbol{w}\left(t-\tau\right)\boldsymbol{u}\left(\tau\right)\,\mathrm{d}\tau, \tag{2}$$

and d_1 and d_2 respectively weight the time required to reach the objective as opposed to the control effort. A minimum

$$\min_{\boldsymbol{u}(t),\boldsymbol{p},T}\left(f\right) \tag{3}$$

is sought.

The formulation of the problem is subjected to the constraints defined by the constrained dynamics of the system expressed in Differential Algebraic (DAE) form for the sake of generality

$$\boldsymbol{F}\left(\boldsymbol{x}\left(t\right),\dot{\boldsymbol{x}}\left(t\right),\boldsymbol{z}\left(t\right),\boldsymbol{u}\left(t\right),\boldsymbol{p}\right) = 0 \tag{4}$$

$$\boldsymbol{G}\left(\boldsymbol{x}\left(t\right),\boldsymbol{p}\right) = 0 \tag{5}$$

where $\boldsymbol{x}(t)$ and $\boldsymbol{z}(t)$ are the differential and the algebraic portions of the state of the system, and $\boldsymbol{p}$ are discrete parameters. The problem may also be subjected to additional equality and inequality constraints $\boldsymbol{C}_{1|2}$

$$\begin{aligned} \boldsymbol{C}_1(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}, T) &= 0 \quad (6) \\ \boldsymbol{C}_2(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}, T) &\geq 0. \quad (7) \end{aligned}$$

The range of validity of the inputs can be subjected to constraints as well

$$\underline{\boldsymbol{u}} < \boldsymbol{u}(t) < \overline{\boldsymbol{u}}. \quad (8)$$

By setting $d_2 = 0$, only a minimization of the time T is sought, regardless of the required effort. This can lead to undesirable results, including instabilities in the system. On the contrary, if $d_1 = 0$, a minimal control power solution is obtained.

A direct approach is used, which computes the values of the inputs $\boldsymbol{u}(t)$ and of the discrete parameters $\boldsymbol{p}$ at each time step in an iterative manner. The inputs $\boldsymbol{u}(t)$ are discretized as

$$u_j(t) = \sum_i q_{ij}(t) \cdot u_{ij} \quad (9)$$

where $q_{ij}(t)$ is an interpolation function and u_{ij} are the values of the j-th input at the i-th interpolation point. Higher-order polynomial interpolation functions yield better interpolation of the control input values; however, when steep gradients occur, higher-order polynomials may result in undesired oscillations that are not physical and may numerically destabilize the optimization. In those cases, a linear interpolation, although coarse, may help in attenuating those undesired effects. Wherever the solution is costant, a zero-order interpolation is used, which allows to reduce the number of unknowns. In the current analysis, both interpolations are used; an adaptation algorithm at the end of each optimization cycle selects what interpolations are most appropriate in a given interval of the analysis time. The parameters $\boldsymbol{p}$ are intrinsically discrete, so p_i indicates the value of the i-th parameter.

The adaptation of the controls time grid is performed by scaling a fixed time step grid by the final time T, which is part of the objective function. The grids of the control inputs are normalized on a scale ranging from 0 to 1; the generic control grid becomes

$$\Gamma : 0 = \tau_0 < \tau_1 < \ldots < \tau_{n-1} < \tau_n = 1, \quad (10)$$

where the i-th interpolation time is $T \cdot \tau_i$. This allows control grids to stretch and shorten as the overall simulation time T changes. The integration time step Δt is constant; it is also scaled to keep the number of time steps constant.

The unknowns of the discrete problem are thus

$$\boldsymbol{y} = \left\{ \begin{array}{c} T \\ \boldsymbol{u}_1 \\ \vdots \\ \boldsymbol{u}_n \\ \boldsymbol{p} \end{array} \right\}. \quad (11)$$

The constraints described by Equations (6, 7) usually express continuous constraint conditions; in those cases they must be discretized. To reduce the amount of data that must be treated

Table 1: Two arm robot optimization results.

Length	1.0	m
Height	0.1	m
Link 1		
Material	Steel	
Thickness	5.0	mm
Mass	3.9	kg
Link 2		
Material	Aluminum alloy	
Thickness	2.5	mm
Mass	0.685	kg

by the optimization procedure, a heuristic algorithm is used to select only those time steps where the constraint is about to be violated. When approaching a constraint violation, the optimization process is interrupted and rearranged to account for the constraints; then it is restarted.

The minimization problem is split in two phases. The first phase addresses the integration of the dynamics of the system to compute the discrete values of the state $\boldsymbol{x}_i$ and $\boldsymbol{z}_i$, which are required to evaluate the objective function (3) and the constraints (6, 7).

This phase is delegated to the multibody analysis, performed with MBDyn as core solver, which computes tentative solutions by "shooting", i.e. forward integrating the problem from tentative initial values and tentative histories of the input commands $\boldsymbol{u}(t)$ and of the discrete parameters $\boldsymbol{p}$.

The use of a general-purpose multibody analysis software makes the selected approach quite general, since it does not pose any drastic limitation on the nature of the problem and on the level of detail and complexity of the analysis. Moreover, this does not require any specific coding of the problem into the optimization procedure, thus saving development time and resources, and at the same time allows a great commonality and reuse of models in the analysis, design, optimization and verification phases. The availability of an Open-Source software eases a tighter integration with the optimization software.

The second phase addresses the optimization problem solution; since all the entities involved in the problem are differentiable, an SQP solver is used. In the present work, the Harwell VF13 routine is used. The gradients of the objective function f and of the constraint functions C_1 and C_2 are numerically computed by means of central differences at all times required by the optimization process. The perturbation of the functions with respect to each variable $\boldsymbol{y}$ is independently performed in parallel on a cluster of computers. The optimization process dispatches the analyses across the cluster by means of the MPI message passing interface.

3 APPLICATION: TWO ARM ROBOT

The robotic arm is a two degree of freedom planar robot as shown in figure 1. The workspace is the horizontal plane, so there are no gravity effects. The two link have the same length (1 meter) but differ in mass and inertia as shown in Table 1.

The robot is controlled by two couples C_1 and C_2 applied to the root and the elbow hinges, respectively, and limited to a minimum and a maximum value of -1 and 1. In order to obtain a coarse approximation of the real system, the hinge angles θ_1 and θ_2 can vary between -135

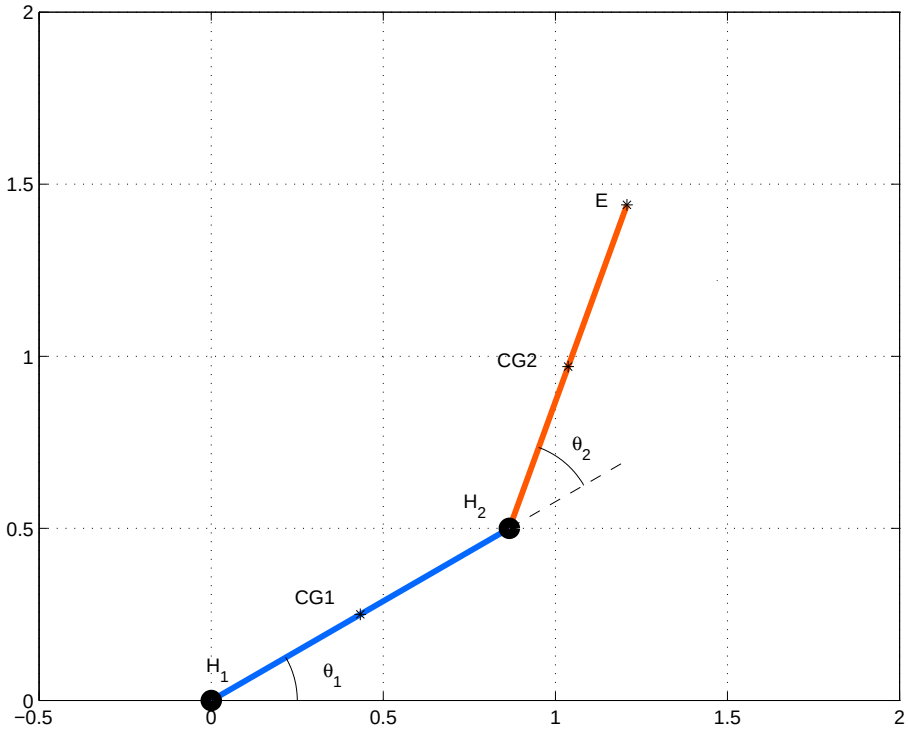

Figure 1: Two arm robot sketch.

Table 2: Two arm robot optimization results.

Iterations	Objective	Variables	Runs	Time
97	4.9605s	12	2425	7m 34s
35	3.4396s	79	5565	17m 23s
28	3.4370s	43	2436	7m 36s
39	3.4377s	49	3861	12m 4s
34	3.4358s	34	2346	7m 20s
Total: 233			16633	52m 0s

and 135 degrees. The optimization goal is to find a path from the starting point, where the end effector is in $\{2, 0\}$, to the final position, where the end effector is in $\{0, 2\}$, that ensures the minimal traveling time. Figure 2 shows the optimal path after four grid adaptations.

The proposed algorithm finds the optimal solution after 233 optimization cycles with four control grid adaptations; further details about convergence are shown in Table 2. The optimization has been carried out on an AMD Athlon 4 GHz PC.

The resulting minimal time is 3.4358s. The weight function used in Equation (2) is essentially a high-pass filter with a cut-off frequency of 8 Hz. If the second hinge is locked, the system reduces to a single rod and the optimal solution results in the root hinge accelerating at the maximum allowed value for the first half of the path, and then decelerating at the maximum allowed negative value for the second part. This acceleration pattern is typical of a bang-bang control strategy. For this solution the traveling time is about 6 seconds. An analogous behavior is obtained from the optimization for the root motor, as illustrated in Figure 3; the figure also shows the torque required by the elbow motor to perform the optimal trajectory.

On the contrary, if the second hinge can rotate, three different effects can contribute in re-

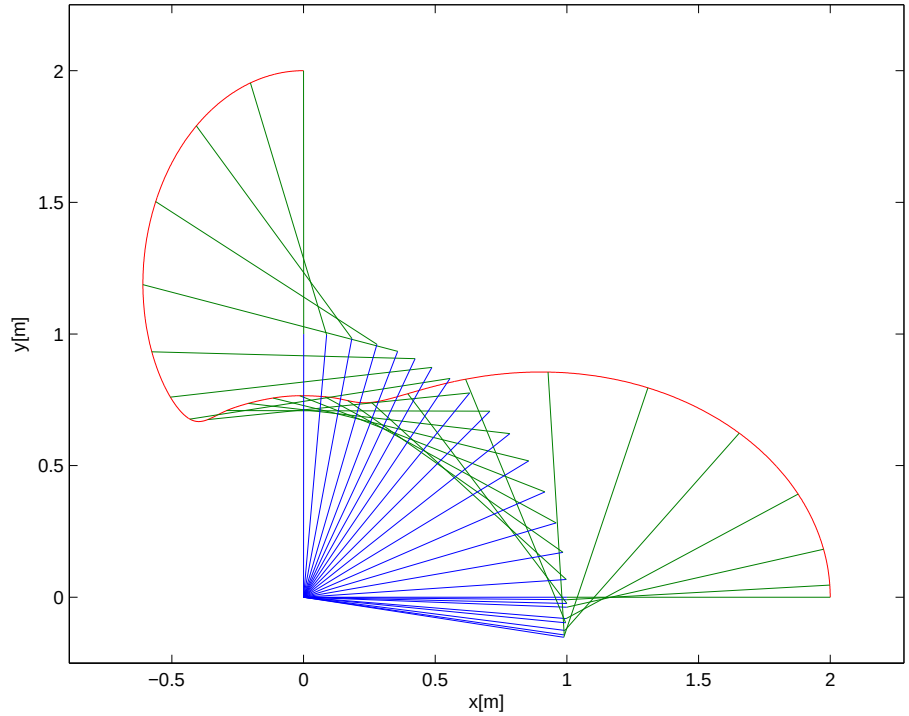

Figure 2: Two arm robot optimal path ($T = 3.4358s$).

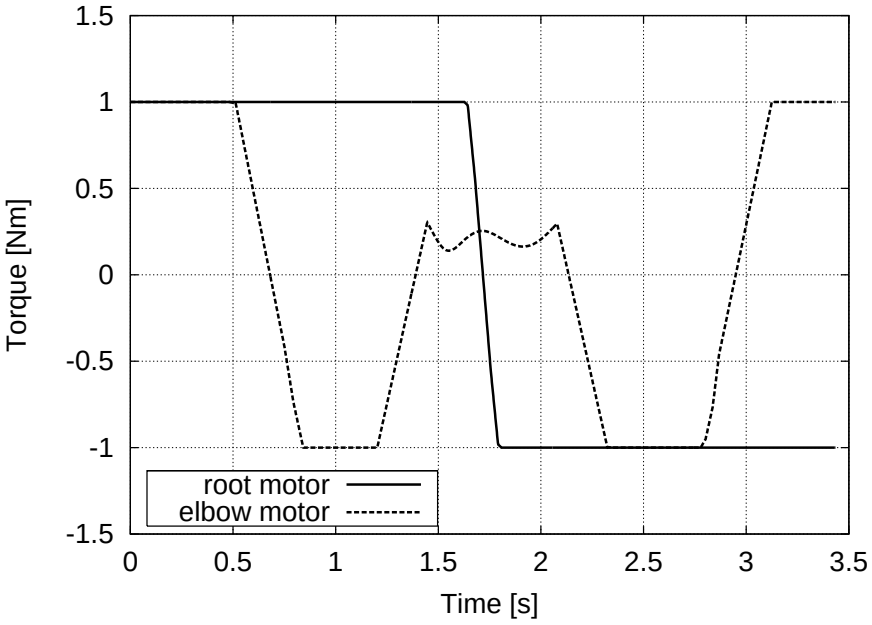

Figure 3: Two arm robot root and elbow motor torques.

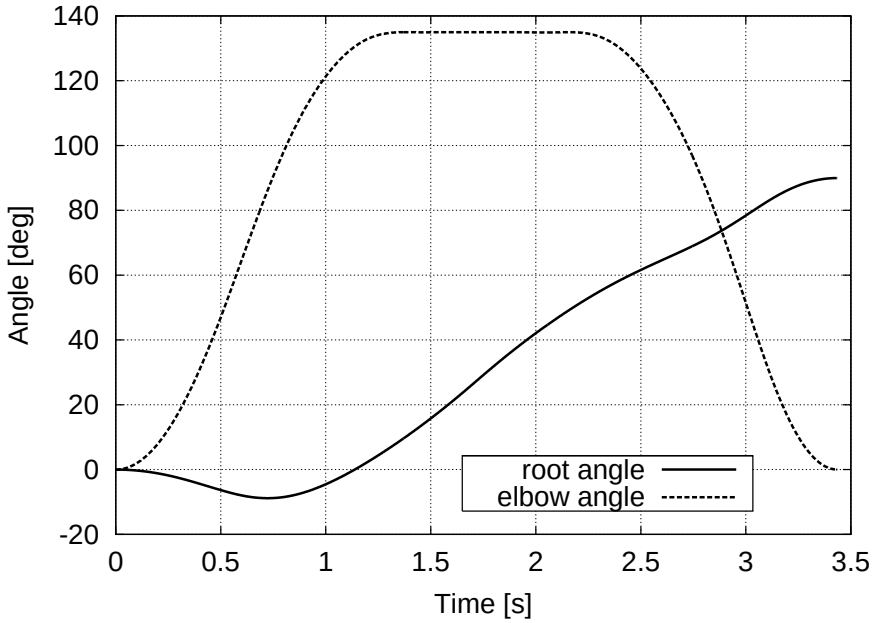

Figure 4: Two arm robot root and elbow angles.

ducing the traveling time: the overall inertia related to the first hinge is decreased, thus allowing higher angular accelerations for a given motor torque; the centrifugal and the Coriolis forces due to the relative motion of the second link with respect to the first one are beneficial because they always provide positive torque values during transients, but a larger value is provided during acceleration and a smaller one, in opposition to the motor torque, is provided during deceleration.

Indeed the optimal path found by optimizer drives the second link forward until the limit elbow angle is reached, in an attempt to reduce the overall inertia associated to the root hinge. The path of the C.G. is clearly illustrated in Figure 6, while Figure 4 illustrates the relative angles of the root and the elbow motors, with Figure 5 zooming on the interval where the constraint on the maximum value (135 deg) is hit.

In principle, from a purely inertial point of view, if no transient to fold the outer arm needs be accounted for, an analogous pattern could have been obtained by initially driving the second link backward, instead of forward, to the limit angle. This solution, however, suffers from an adverse effect of the inertia during the folding transients, opposed to the favorable one that is clearly illustrated in Figure 7, which shows the apparent torque terms on the root motor associated to the centrifugal and Coriolis forces that occur during the elbow angle transients.

During the initial acceleration, the apparent torque contributes to the acceleration of the root arm; during the final deceleration, the apparent torques contrast the torque of the motor, but their amplitude is reduced with respect to their value during the initial phase. This behavior is reported in the literature [2].

The optimization process seems to be quite robust, because even if the initial conditions are perturbed in order to force the solution in the direction of the opposite solution, with the root arm leading forward and the outer arm lagging behind, it always converges to the same optimal solution.

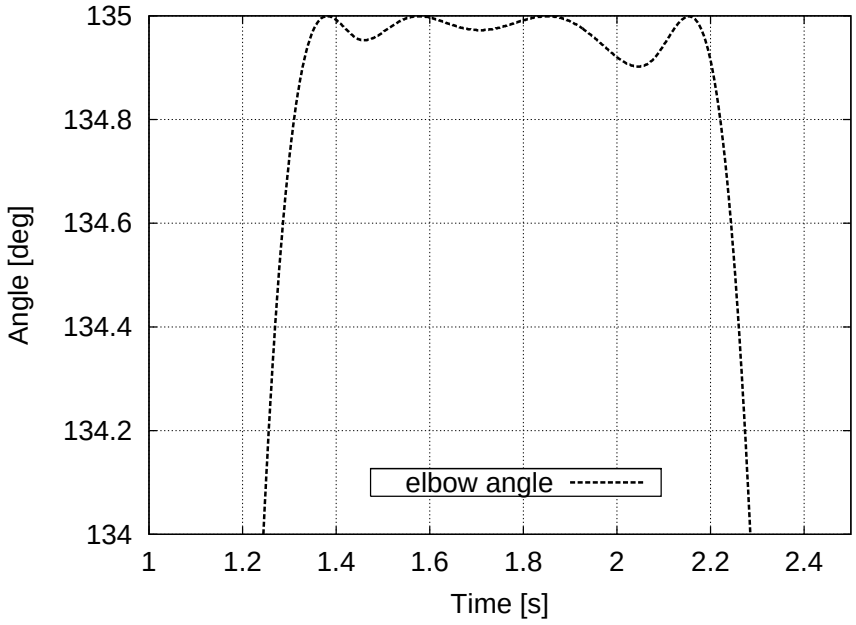

Figure 5: Two arm robot elbow angle where the constraint on the maximum value is hit.

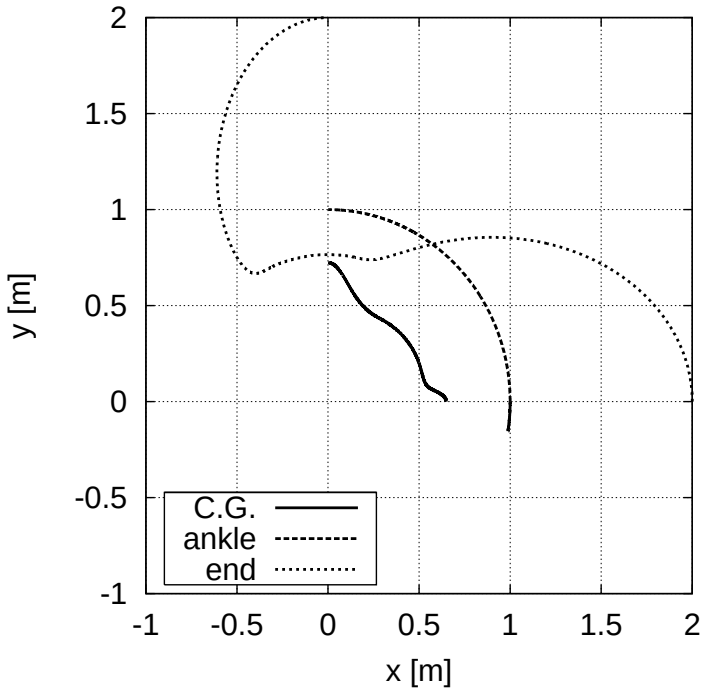

Figure 6: Two arm robot C.G. position during motion.

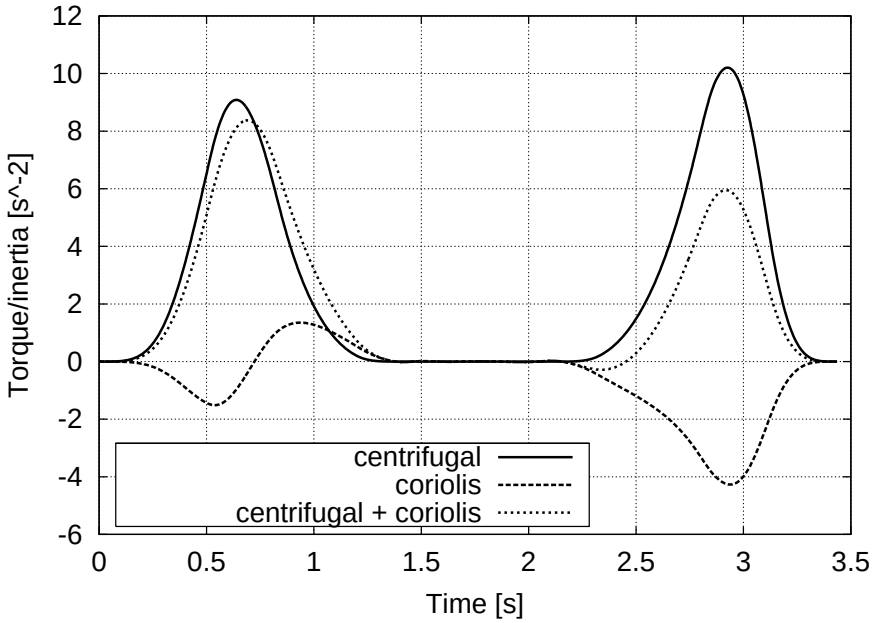

Figure 7: Centrifugal and Coriolis apparent torques seen by the root motor of the two arm robot.

4 REAL-TIME SIMULATION

A distinguishing feature of MBDyn is represented by its versatility, which allows to use the same software and almost the same model, except for selected switches to control the execution of the simulation, to perform both the trajectory optimization and the real-time simulation tasks.

The real-time capability has been developed to investigate the feasibility of real-time enabling a general-purpose simulation software, thus allowing arbitrary details in the modeling of mechanical and multidisciplinary problems, as soon as they comply with the stringent scheduling requirements of real-time simulation.

Two self-imposed constraints drove the design and implementation:

- preserve as much commonality as possible between the real-time and the batch simulation scheduling;
- allow real-time simulations to be run on low-end, off-the-shelf hardware (typically, single/dual ix86 PCs).

In order to give a rough idea of the current performances of the software, a 6 DOF robot, with selected deformable components, including friction modeling at the revolute joints, resulting in 120 equations, runs at 2 kHz on a stock 2.4 GHz Athlon [6]. A rotorcraft wind-tunnel model, with selected deformable components, aerodynamic blade elements and complete control system kinematics, resulting in 180 equations, runs at approximately 800 Hz on the same hardware; early results presented in [3] indicate much lower figures, which were recently dramatically improved by introducing the new specialized linear solver described in [5].

The two-arm problem, with the optimal controls, has been simulated in real-time in [1] on a dual 1.7 GHz Athlon with encouraging results: the rigid robot simulations could be easily run with a sample frequency of 1 kHz, which is about twice the sample rate used by the present optimization process. More recent analyses, exploiting the software improvements illustrated in [6] and run on slightly faster hardware (2.4 GHz Athlon), show a sampling rate about 5 times

higher, which promises to allow the electric motor dynamics modeling within the multibody simulation. The sample rate of 1 kHz could be easily preserved even when deformable arms are modeled by beam elements; this result could not be achieved in [1], where the highest sampling rate allowed by the deformable robot was 330 Hz.

The investigation of the robustness of the optimized control strategy with respect to disturbances and to model refinement, e.g. the above discussed introduction of the deformability of the arms, will be the subject of further investigation.

REFERENCES

[1] M. Attolico and P. Masarati. A multibody user-space hard real-time environment for the simulation of space robots. In *Fifth Real-Time Linux Workshop*, Valencia, Spain, November 9–11 2003.

[2] G. Giese, R. W. Longman, and H. G. Bock. Mechanical assessment of time optimal robot motion. *Computational Mechanics*, 33:121–128, 2004.

[3] P. Masarati, M. Attolico, M. W. Nixon, and P. Mantegazza. Real-time multibody analysis of wind-tunnel rotorcraft models for virtual experiment purposes. In *AHS* 4^{th} *Decennial Specialists' Conference on Aeromechanics*, Fisherman's Wharf, San Francisco, CA, January 21–23 2004.

[4] P. Masarati, M. Morandini, G. Quaranta, and P. Mantegazza. Open-source multibody analysis software. In *Multibody Dynamics 2003, International Conference on Advances in Computational Multibody Dynamics*, Lisboa, Portugal, July 1–4 2003.

[5] M. Morandini and P. Mantegazza. Using dense storage to solve small sparse linear systems. Submitted to *ACM Transactions on Mathematical Software* (ACM TOMS).

[6] M. Morandini, P. Masarati, and P. Mantegazza. A real-time hardware-in-the-loop simulator for robotics applications. In *Multibody Dynamics 2005, ECCOMAS Thematic Conference*, Madrid, Spain, June 21–24 2005.