7th ESA Workshop on Advanced Space Technologies for Robotics and Automation 'ASTRA 2002'
ESTEC, Noordwijk, The Netherlands, November 19 - 21, 2002

Dynamics Simulation and Assembly Environment for Rapid Manipulator Design

Rainer Krenn, Bernd Schäfer, Gerd Hirzinger

Institute of Robotics and Mechatronics
German Aerospace Center, DLR, D-82234 Wessling, Germany
E-mail: rainer.krenn@dlr.de, bernd.schaefer@dlr.de, gerd.hirzinger@dlr.de

Abstract

Currently, DLR is developing its 3rd generation of light-weight robots in modular design. This process includes the development of high performance drives, carbon fiber arm structures and novel wrist units. The robotics systems are dedicated for applications on ground as well as for future space missions. The robotic components are stored in a database that can be accessed via an assembly panel inside a simulation environment and enables a system engineer to rapidly design a fully operating concept of a robotic system. One of the design tasks is the definition of the robot kinematics, respectively the number of joints and the sequence of joint axis orientations. For space applications redundant kinematics with at least 7 joints are preferred due to their increased skill performance and flexibility. However, the problem of solving the inverse kinematics increases accordingly to the number of joints. Currently a Lagrange constraint optimization and a method using a differential equation system are provided.

INTRODUCTION

Since two years, DLR is developing its 3rd generation of light-weight robots, called LBR 3 [1] (Fig. 1). A major design goal thereby is the modularity of the robot components in order to provide a construction kit for a rapid manipulator setup. In parallel a software environment for virtual assembly, modeling and simulation of robotics systems based on that components has been created. In the paper the architecture and implementation of the assembly, modeling and simulation environment is presented and a general solution for the specific problem of inverse kinematics of kinematically redundant robots is pointed out.

LIGHT-WEIGHT ROBOT COMPONENTS

The activities concerning light-weight robotics inside the Institute of Robotics and Mechatronics cover the development of high performance drives, carbon fiber arm structures and novel wrists units (Fig. 2). Moreover, dexterous multi-fingered hands are an essential part of the robotics technology of the institute.

Fig. 1. DLR's 3rd Generation of Light-Weight Robots

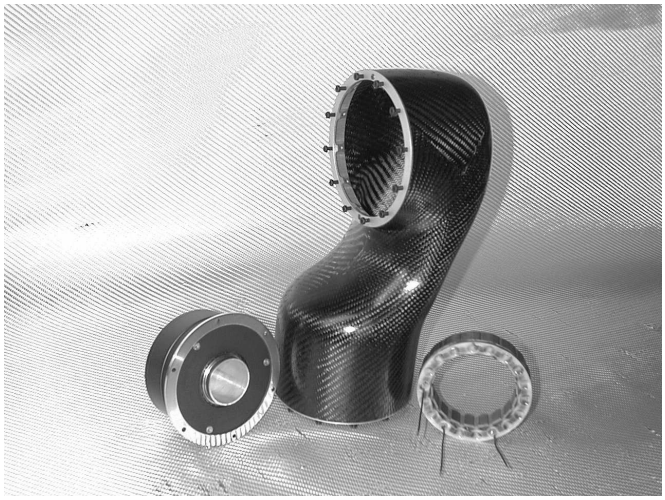

Fig. 2. Components of LBR 3

The drives are compact units consisting of brushless DC multi-pole motors and light-weight Harmonic Drives together with an electromagnetic safety brake (Fig. 3). The decisive improvement of the drive power was the consequential adaptation of the motors to the performance (gear ratio and maximum torque) of the attached Harmonic Drives. As a result of these developments an optimum of power loss and weight reduction was found.

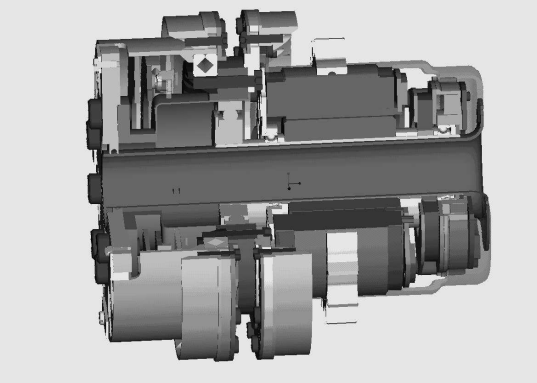

Fig. 3. Drive Unit of LBR 3

The novel arm structures consist of carbon fiber with integrated aluminum retainer to attach the drives to (Fig. 2). The weight of such kind of link is negligible compared with other components of the manipulator. The links are designed in different shapes like L-shape, C-shape, S-shape etc. in order to realize an almost infinite number of robot kinematics. Moreover, two types of design lines provide the option to assemble robots with so-called symmetrical (roll joint axis inside the pitch joint plane) and asymmetrical kinematics (Fig. 4).

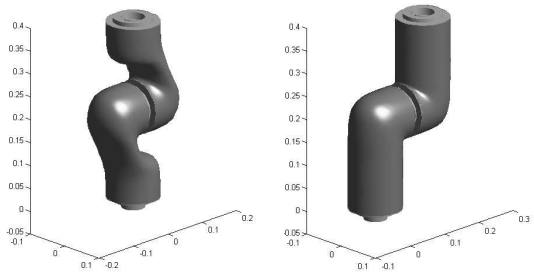

Fig. 4. Symmetrical and Asymmetrical Links

The wrists of that robot generation are realized by highly integrated roll-pitch-roll joint units. The characteristic component is the ball like housing where the last two joints are assembled. The wrists are available with a plane end-effector or an additional bracket in order to provide a kind of gimbaled end-effector kinematics (Fig. 5).

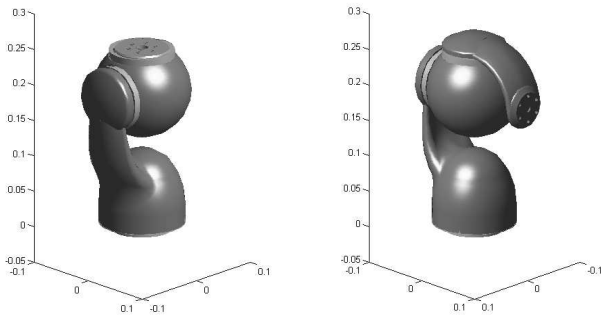

Fig. 5. Wrist Units of LBR 3 with Roll-Pitch-Roll and Gimbaled Kinematics

The robotics systems that are based on LBR 3 technology are dedicated for applications on ground (e.g. medical technology) as well as for future space missions. Currently the components of LBR 3 are used in different scenarios like

- in the manipulator inspection system (MISSISS) as a contribution to the International Space Station ISS [2] shown in Fig. 6,
- in satellite servicing missions, e.g. ROSAT capturing and de-orbiting [3] (Fig. 7),
- or as basic system for our national robotic component verification experiment on ISS, called ROKVISS [4] (Fig. 8).

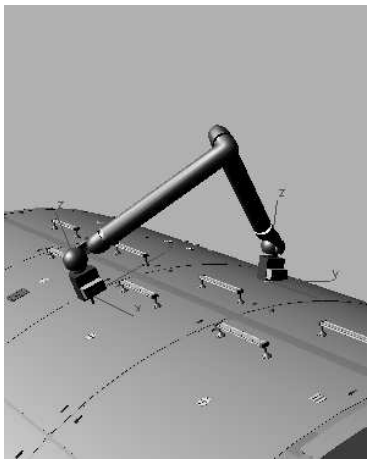

Fig. 6. MISSISS scenario

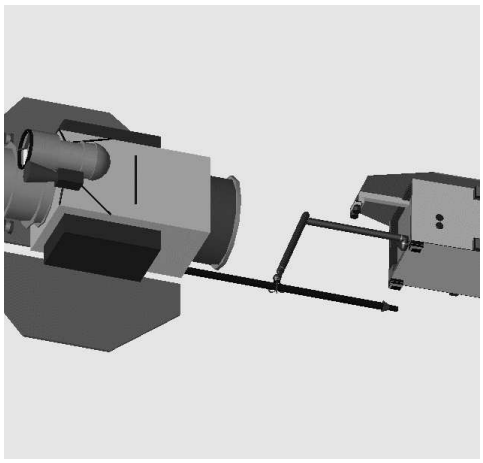

Fig. 7. ROSAT Capturing

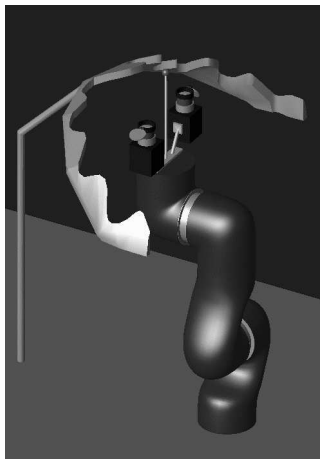

Fig. 8. ROKVISS

ASSEMBLY, MODELING AND SIMULATION ENVIRONMENT

In order to enable a system engineer who is working on such kind of robotic scenarios to rapidly assemble a manipulator prototype and to build a fully operating kinematics and dynamics simulation of that prototype, the components of LBR 3 respectively their data are stored in a database. Using our specific assembly, modeling and simulation environment the database can be explored and a robot model can be assembled using the LBR 3 database components.

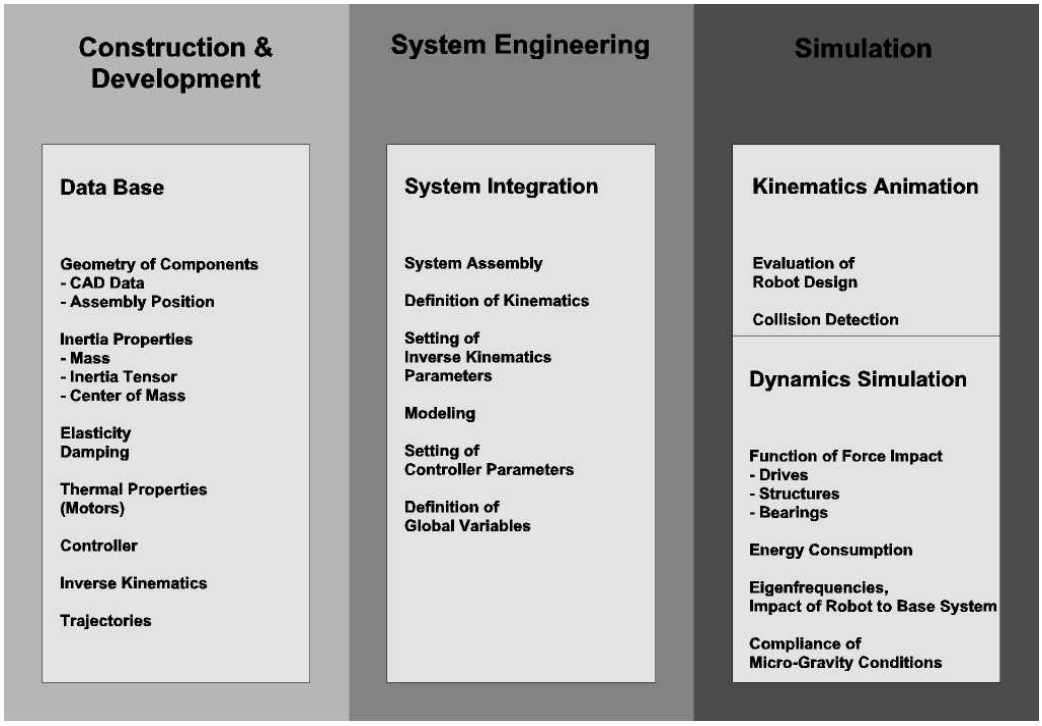

Fig. 9. Layers of Robot Assembly, Modeling and Simulation

The environment is divided into three layers (Fig. 9):

- The Construction & Development layer,
- the System Integration layer and
- the Simulation layer.

The implementation of layer specific tasks in software and the used software packages for that tasks are outlined in Fig. 10.

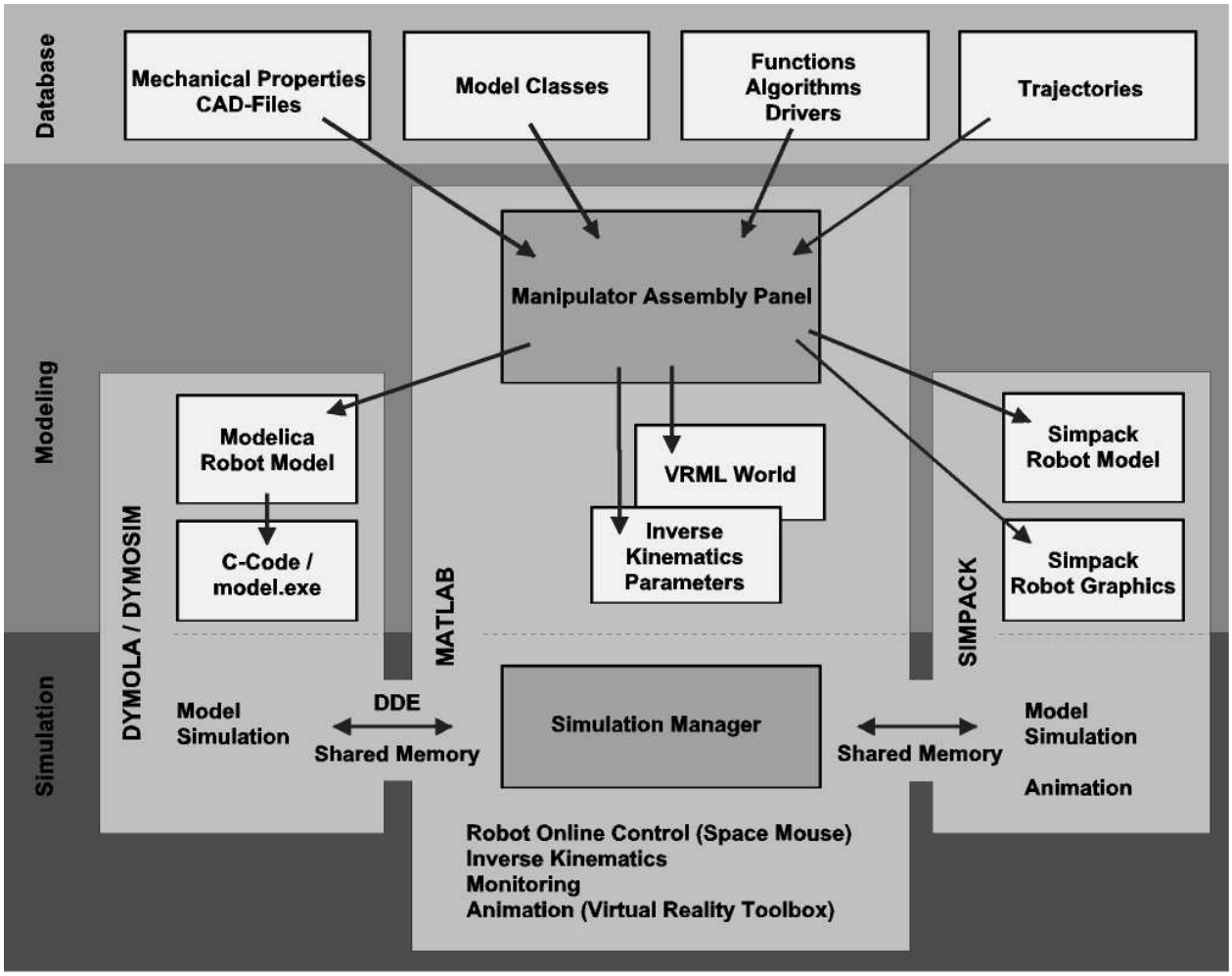

Fig. 10. Software Implementation of Assembly, Modeling and Simulation Environment

Construction & Development Layer

As already mentioned, the first layer, the Construction & Development layer, results in a database of all components of the robotic system. It includes different kind of libraries:

- The first one are libraries that are dedicated to single robot components like links, motors and gears. These are graphics information of the component surface shape (CAD files) and files of physical properties of the components like mass, inertia tensor, center of mass, force/torque limits, gear ratio, stiffness, damping, thermal properties etc.
- The second kind of library is a collection of model classes of the robot component types (links, joints, joint controller, etc.). They describe in an object oriented manner the dynamics behavior of the components. In the modeling phase, an instance of them will be taken for each component of the assembled robot and will be parameterized according to their physical data. Due to the simulation software that is used inside our environment, DYMOLA and SIMPACK, the model classes are stored in the respective languages MODELICA and SIMPACK code.
- A third group of libraries are functions that affect the total robotic system. In our environment these functions are drivers for different user input devices like the Space Mouse 6D controller or an haptic interface, and inverse

kinematics algorithms for global robot control. Currently two different solutions of the inverse kinematics problem are provided inside the presented environment. The first one is a Lagrange constraint optimization, which provides the possibility to minimize a function, mostly a quadratic expressions in joint position, speed, acceleration or torque subject to the end-effector misalignment. A second and more uncommon way of inverse kinematics implementation is to make use of the differential equations system of the robot motion whereby the robot is modeled as a passive chain of joints and limbs. A detailed description of inverse kinematics implementation is given in the chapter Inverse Kinematics for Redundant Manipulators.

- Another group of libraries are static tables that contain predefined trajectories and test routines for different applications.

System Integration Layer

The objects of the database can be accessed and assembled to a robotic system inside the System Integration layer. The software implementation of this layer is a MATLAB based graphical user interface called Manipulator Assembly Panel (Fig. 11).

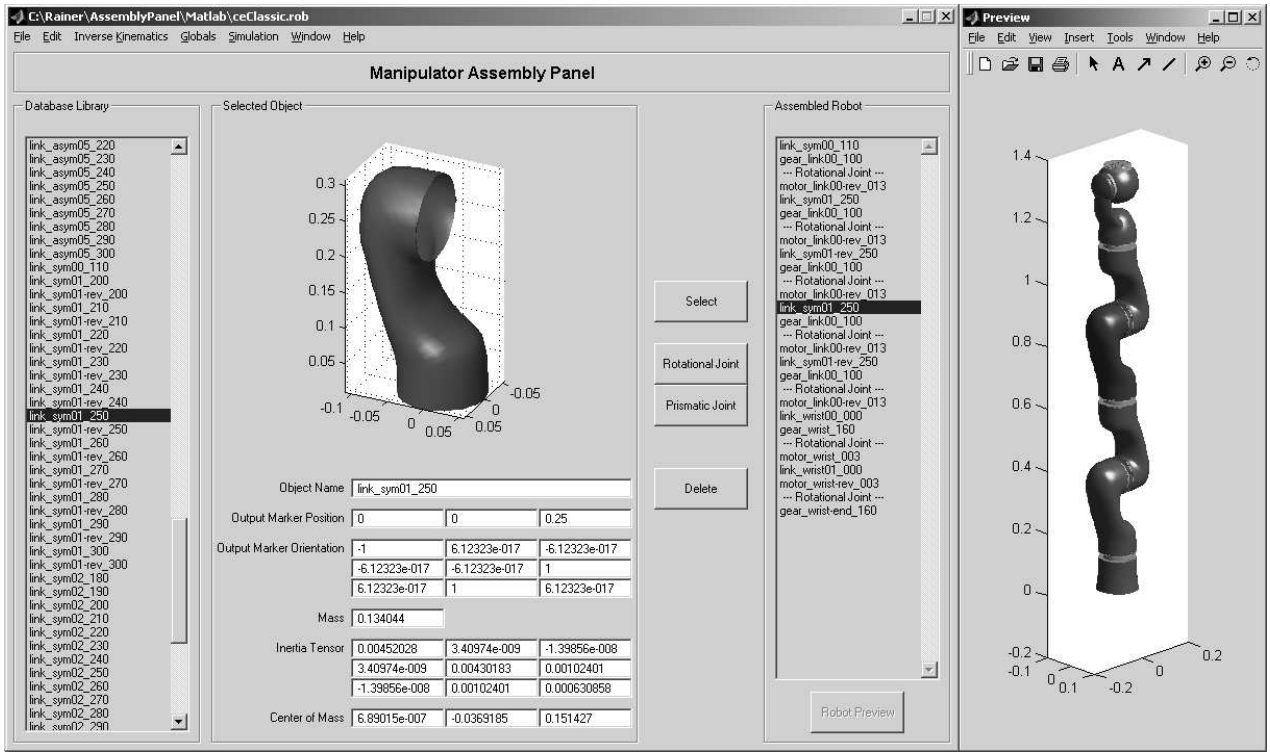

Fig. 11. Manipulator Assembly Panel

Data Import
The first phase of system integration is the data import phase. Herein, single robot parts or pre-assembled units from the LBR 3 database can be connected to a complete robotic system just by mouse clicks. The user will be supported by graphical and alpha-numerical information about the components and by a preview of the assembled system.

Model Export
In the model export phase the robot will be provided, again by mouse click, in different descriptions that are required for subsequent simulations. The main sections that need their own robot descriptions are inverse kinematics, dynamics simulation with robot control and graphical animation. Thereby, an important item is the consistency of all exported code. This will be guaranteed by the central modeling environment that generates the code.

Inverse Kinematics:
We assume that for control purposes the robot can be represented by a chain of rigid bodies that are connected by joints with one degree of freedom each. Therefore, the required data to parameterize the inverse kinematics function (Lagrangian optimization algorithm) are

- the number of joints,
- the joint type, linear or rotational,
- the distance vectors between the assembly points of a link, called input marker and output marker,
- the rotation matrix between input marker and output marker,
- the weighting factors of the optimization criteria for the robot joint motion.

Dynamics Simulation:
For simulation purposes the robot model including its joint controller will be exported in MODELICA code and SIMPACK code. In both options the robot model code consists of objects (parameterized instances) of classes taken from the database and information about connections between the objects. Additionally, the models will be equipped with interfaces for simulation data exchange like global end-effector command input and system state variables output.
For a subsequent simulation run, the SIMPACK model can be taken as it is. It will be processed inside the SIMPACK simulation environment. However, the MODELICA code must be translated into C-code using DYMOLA and compiled and linked with specific libraries afterwards. The compilation result is a stand-alone simulation application of the robot, called DYMOSIM, that runs independently from specific simulation software packages.

Graphics Animation:
For graphics animation of the robot motion, again two different options are supported by the Assembly Panel: The first option is the SIMPACK 3D graphics animation. Choosing this option the Assembly Panel exports a robot shape description in SIMPACK code as well as SLP render files of the robot link units (e.g. output side of joint i, arm structure, input side of joint $i+1$) to the SIMPACK CAD-geometry database. Using these files the SIMPACK simulation environment can provide a robot animation in parallel to the running robot simulation.
The second option is the export of a VRML robot shape description. Due to the standardization VRML this option offers a more independent way for graphics animation.

Simulation Layer

The Simulation Layer of the Assembly, Modeling and Simulation Environment covers the command generation, the dynamics simulation (time integration) and the kinematics animation of the robot model and the data monitoring. Inside this layer the robotic system will be verified and validated. The simulations can run in non-real-time as well as in real-time. This option is essential, if online control is provided by the simulation environment.

Command Generation:
The architecture of our environment associates the command generation with the central MATLAB based Simulation Manager. Hereby, command generation means online-control using the Space Mouse 6D input device and subsequent generation of joint motion commands by the inverse kinematics algorithm. The transfer of joint motion commands to the simulation modules is realized by Dynamic Data Exchange (DDE) with DYMOSIM only or via shared memory with DYMOSIM and SIMPACK.

Dynamics Simulation:
The time integration will be performed inside DYMOSIM or SIMPACK environment under control of the central Simulation Manager. Analogously to the command input, the state variables output takes place via DDE from DYMOSIM only or via shared memory from DYMOSIM and SIMPACK.

Kinematics Animation:
The 3D animation of the robot motion runs in two different setups. The first one uses the animation capabilities of the SIMPACK environment. However, this is only possible, if the simulation itself is running in the SIMPACK environment, too. The second setup uses VRML capabilities together with a JAVA based server that controls the animation in a VRML compatible viewer (e.g. HTML browser). The implementation of this setup is done using MATLAB's Virtual Reality Toolbox functions.

Data Monitoring:
The monitoring of state variables and parameters of the running simulation is implemented using MATLAB's graphics capabilities. The variables to be shown can be selected by mouse click in a Variable Browser.

INVERSE KINEMATICS FOR REDUNDANT MANIPULATORS

One of the major robot design tasks is the definition of the robot kinematics, respectively the number of joints and the sequence of joint axis orientations. For space applications redundant kinematics with at least seven joints are preferred due to their increased skill performance and flexibility. However, the problem of solving the inverse kinematics increases accordingly to the number of joints. Additionally, the inverse kinematics has to support online interactive control by user input without any command pre-processing. Currently two different solutions are provided inside the realized environment.

Lagrangian Optimization

The first solution is a Lagrangian constraint optimization (1),

$$L = f(\mathbf{q}) + \boldsymbol{\lambda}^T \cdot \mathbf{h}(\mathbf{q}) \quad \rightarrow \quad \min \tag{1}$$

which provides the possibility to minimize a function subject to the end-effector misalignment (2),(3).

$$\boldsymbol{\Delta} = \mathbf{B}_{desired}^{-1} \cdot \mathbf{B}_{robot} \tag{2}$$

$$\mathbf{h} = \begin{pmatrix} \Delta_{14} \\ \Delta_{24} \\ \Delta_{34} \\ \Delta_{12} - \Delta_{21} \\ \Delta_{23} - \Delta_{32} \\ \Delta_{31} - \Delta_{13} \end{pmatrix} = \mathbf{0} \tag{3}$$

The function to be minimized consists of quadratic expressions in joint position (4), speed (5), acceleration (6) or drive power (7). Depending on the preferences the influence of the minimization goal can be weighted by different factors given by the matrix *P* in (4) to (8).

$$f_{\mathbf{q}} = (\mathbf{q} - \mathbf{q}_{ref})^T \mathbf{P}_{\mathbf{q}} (\mathbf{q} - \mathbf{q}_{ref}) \tag{4}$$

$$f_{\dot{\mathbf{q}}} = \dot{\mathbf{q}}^T \mathbf{P}_{\dot{\mathbf{q}}} \dot{\mathbf{q}} \tag{5}$$

$$f_{\ddot{\mathbf{q}}} = \ddot{\mathbf{q}}^T \mathbf{P}_{\ddot{\mathbf{q}}} \ddot{\mathbf{q}} \tag{6}$$

$$f_{\mathbf{p}} = \mathbf{p}^T \mathbf{P}_{\mathbf{p}} \mathbf{p} \tag{7}$$

In space robotics often the force/torque impact (8) from the robot base to the support system (satellite) has to be minimized in order to guarantee micro-gravity conditions or attitude control.

$$f_{\mathbf{F}_0} = \mathbf{F}_0^T \mathbf{P}_{\mathbf{F}} \mathbf{F}_0 \tag{8}$$

Solution of a Differential Equations System

A second and novel way of inverse kinematics implementation is to make use of the differential equations system of the robot motion whereby the robot is modeled, a second time, kinematically identical to the robot to be moved but as a passive chain of joints and limbs (Fig. 12).

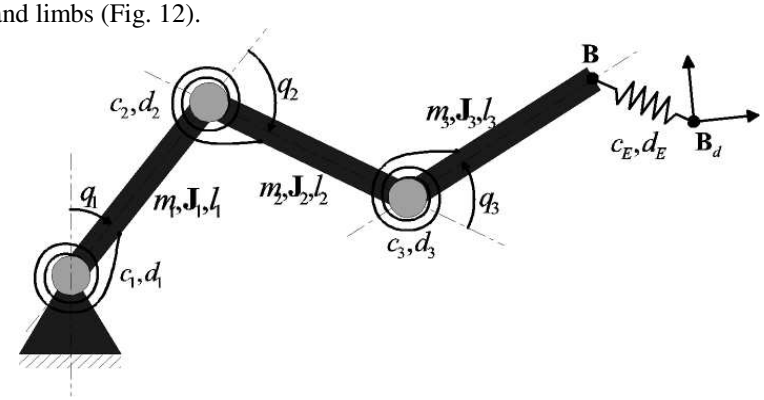

Fig. 12. Robot Modeling as Passive Chain for Inverse Kinematics Purposes

In this case the motion is initiated by spring force and torque that is a function of a virtual spring stiffness and the misalignment between the end-effector position and the desired position. Then, the joint motion can be calculated by solving the corresponding differential equations. The resulting joint motion of this equation system is now the joint motion command of the simulated robot.
Since the robot is a kinematically redundant one, variations of the dynamics parameters of the kinematics chain (e.g. joint friction, limb mass) affect the overall robot motion. By suitably adjusting these parameters the motion can be optimized according to a particular goal analogously to the optimization inside the Lagrange algorithm. This method is very appealing because it can be easily implemented in available multi-body dynamics simulation software.
In order to use this method online the integration time has to be minimized. This can be achieved by an optimized distribution of limb mass/inertia and joint stiffness/damping inside the robot chain. Assuming the robot chain as a linear system (9) with springs and dampers,

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{D}\dot{\mathbf{q}} + \mathbf{C}\mathbf{q} = \mathbf{F} \tag{9}$$

the parameters have to be set in a way that the mass/inertia of the end-effector body respectively the stiffness/damping of the spring/damper between the end-effector and the target position are some orders of magnitude higher than all the rest which have almost identical values each. In this configuration the system modes almost agree (Fig. 13) and the system achieves only one dominant eigenmode (Fig. 14).

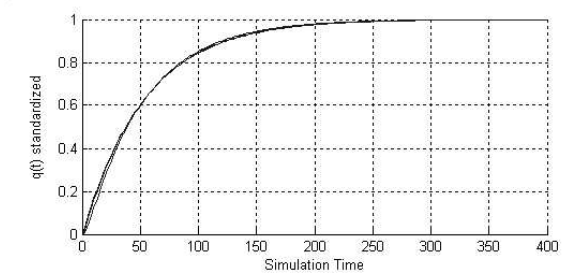

Fig. 13. Similar Standardized Modes

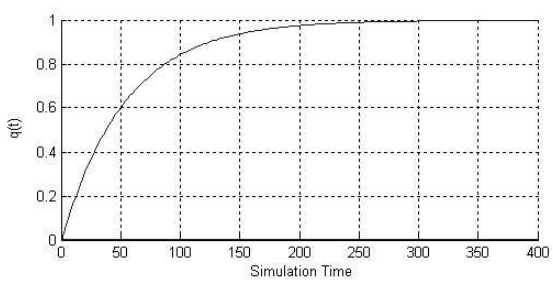

Fig. 14. Single Dominant Mode

If the corresponding eigenfrequency is damped strongly enough, the eigenmotion reaches a tolerance band around its final position in a predefined simulation time. From the simulation point of view this system can be integrated very fast using an implicit integration method with wide tolerances and long step sizes in order to obtain the desired result in a very short computing time.
Important advantages of this method are the robustness close to kinematics singularities and the possibility to command goal positions out of the robot workspace whereby the robot motion results in the most close configuration to that command. Therefore, the method can be applied for systems with less than 6 degrees of freedom, too.

CONCLUSION

The motivation for building our assembly and simulation environment was the rapid design idea supporting the development of robotic-based mission scenarios already in the early design phase. The overall performance of the environment has been tested by applying it to space robot mission examples like in ISS-based and satellite capturing scenarios. The implementation of the environment is not finally finished yet. It will be constantly updated with the latest developments of our institute concerning light-weight robot components.

REFERENCES

[1] G. Hirzinger, A. Albu-Schäffer, M. Hähnle, R. Krenn, A. Pascucci, M. Schedl, N. Sporer, “DLR’s torque controlled light weight robot III – are we reaching the technological limits now?”, *International Conference on Robotics and Automation 2002*, Washington D.C., 11-15 May 2002
[2] B. Schäfer, R. Krenn, G. Hirzinger, A.R. da Silva, “Light-Weight Space Robotics: Rapid Design Approach and Efficient Simulation Environment”, *Machine Intelligence & Robotic Control*, Vol. 3, No. 3, Page 99 – 111 (2001)
[3] OHB System, DLR; “Operationeller Servicing Satellit: OSS Machbarkeitsstudie”, *Final Presentation*, Bonn, Germany, 27 Sept. 2000
[4] http://www.weblab.dlr.de/onl/article.asp?ID=146