Simulation of the on-orbit construction of structural variable modular spacecraft by robots

Matthias J. Reiner[1]

[1]German Aerospace Center (DLR), Institute of System Dynamics and Control (SR), Wessling, Germany,
Matthias.Reiner@dlr.de

Abstract

This paper gives an overview on the simulation of the on-orbit construction of structural variable modular spacecraft by robots using Modelica. For this purpose, a new concept using so-called tensor bodies was developed which enables the fast and continuous simulation of complex scenarios even during structural changes. This research was part of two ESA and EU projects. An overview of the modeling and simulation approach will be given. The scenarios include the on-orbit re-configuration of a modular satellite with a walking robot manipulator and the construction of a modular space antenna array platform with a walking robot with two arms and a torso.
Keywords: Modelica, variable structure, walking robot, on-orbit servicing

1 Introduction

Future advanced spacecraft and orbital platforms can require on-orbit assembly either because of the size of the structures (e.g. large antennas, solar facilities or telescopes) or because of a desired modularity. Since on-orbit human labor is extremely expensive and dangerous, on-orbit assembly and servicing tasks should be done (mostly) autonomously by robots. Because of the complexity and cost of these types of missions, simulation and demonstrator studies on earth should be performed before committing necessary resources for a real mission. Recent research projects by ESA and the EU are working on this topic and DLR-SR was involved in the modeling and simulation of these complex scenarios.

Within the Horizon 2020 EU-funded project MOSAR (Modular and Re-Configurable Spacecraft, see (Letier, Yang, et al. 2019)) a ground demonstrator for on-orbit modular and re-configurable satellites as well as the supporting software and control system was developed.

The basic idea of the project is to use a set of re-usable spacecraft modules as part of a global eco-system. Each individual module can be dedicated to a specific function such as control, power, thermal management or sensors. Once assembled, they will allow the full functionality of the spacecraft. A symmetric walking robotic manipulator (WM, see (Deremetz, Letier, et al. 2020)) allows to capture, manipulate and position the spacecraft modules, while being able to reposition itself on the standard interfaces (abbreviated as SI or Hotdock, see (Letier, Siedel, et al. 2020)) of the spacecraft or on the modules. These SIs provide mechanical, data, power and thermal transfer for interconnection between the modules, spacecraft and the walking manipulator. They are actuated to ensure a safe connection when closed and are containing tubes for the heat transfer as well as connector ports for data and electricity. Within MOSAR, DLR-SR was responsible for the development of a Functional Engineering Simulation (FES) environment and design tool, offering assistance for module design, system configuration and operation planning, with the support of a multi-physics engine.

The FES as well as the design tool were developed in Modelica with an additional MATLAB interface for the project partners. Fig. 1 shows an example for the graphical output of the FES using DLR SimVis (see (T. Bellmann 2009), (Hellerer, Tobias Bellmann, and Schlegel 2014) and (Kümper, Hellerer, and Tobias Bellmann 2021)) for one of the simulated on-orbit scenarios, which is generated directly during the simulation by the Modelica model. The visualization displays a simplified representation of the most important components of the MOSAR scenario. The colored box blocks represent the individual modules. The pipes within the modules represent the (possible) flow of power and heat within the modules. The spheres within the modules are used to represent the powered state of the module (symbolized by a green or red star in the middle) and the color of the sphere represents the current temperature. The seven-axis robot WM is shown in grey The visualization was imported from CAD data provided by SpaceApplications.

In a similar (still ongoing) ESA project call MIRROR (Multi-arm Installation Robot for Readying ORUS and Reflectors), the on-orbit assembly of a large reflector consisting of hexagonal shaped modules is investigated using a ground equivalent laboratory demonstrator. For this purpose, the novel Multi-Arm Robot (MAR) is used which is a modular robot composed of three robotic subsystems: a torso and two symmetrical 7-degree of freedom (DOF) anthropomorphic arms. The arms are based on the WM design from the MOSAR project. The torso has one additional DOF and is the main body of the robot. This mechanical hub can equip three other appendages (limbs or payloads) or can be attached directly to the spacecraft structure (see (Deremetz, Grunwald, et al. 2021)) using

 10.3384/ECP2118638 DOI

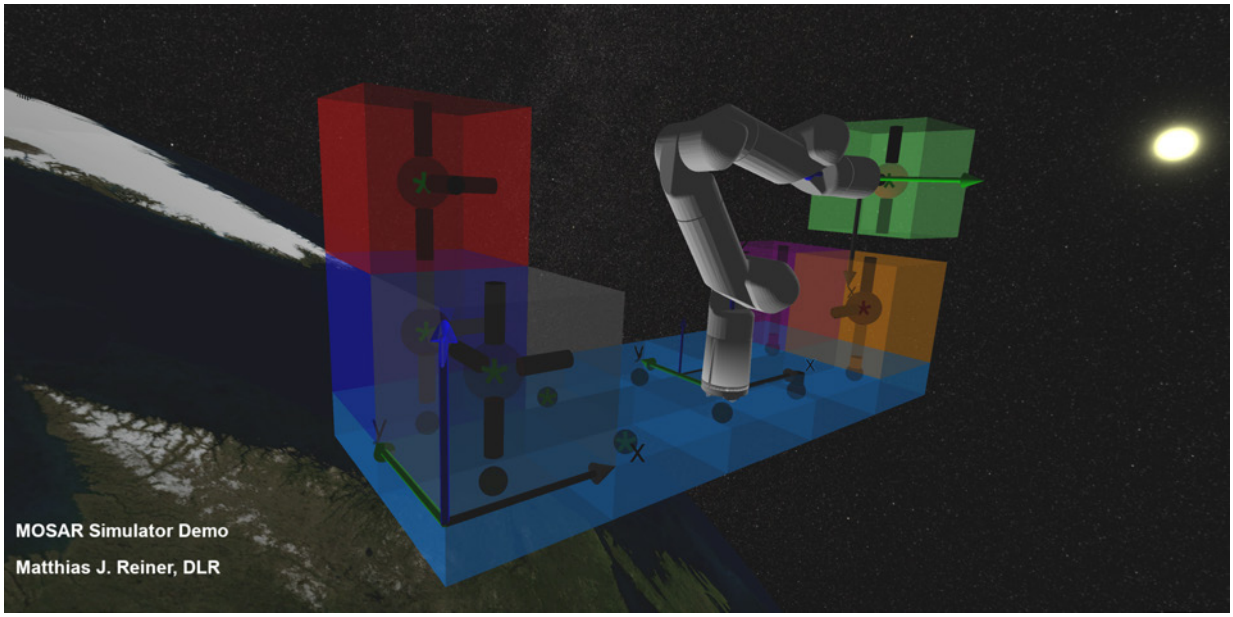

Figure 1. Visualization generated by the FES for an on-orbit assembly scenario investigated in the MOSAR project.

SIs. In this project, DLR-SR is responsible for the simulation of the scenario as well. A simulation tool called MIRROR Kinematic Dynamic & Graphical Simulator (abbreviated in the following as KDG) was developed, which extends the FES from the MOSAR project and is also developed in Modelica with an additional MATLAB interface. Fig. 2 shows the visualization generated by the KDG (also using SimVis). The hexagonal elements, which are used to build up the array, are taken from a stack on the servicer satellite (simplified represented by the yellow bar connector) by the MAR. In the configuration shown in the figure, the MAR is connected to a module with its SI at one of its WM arms and places another module at the edge of the array. In this configuration it has a very long reach. Alternatively, the MAR can also be connected to the array with its torso to enable handling with two arms simultaneously.

The focus of both projects is different, but from a simulation standpoint they share a lot of similarities. The focus for both simulation tools is to get insight in the dynamics of the systems at an early development stage and to test control algorithms for the robots and the logic and handling of the modular components and the standard interfaces, which are used to connect the elements as well as to allow the robots to move over the modules. To enable this the robots have the same SI connectors at the tip of the arms and torso. This allows the MOSAR symmetric walking manipulator (WM) to move by connecting one end of the robot to an SI and then with the other end to another (and switching the working base in the process). The MAR with its two arms and torso has even more possibilities to move in a similar manner.

The SI connectors can not only provide a mechanical connection, but can also be used to transfer power and exchange heat, what also has to be considered for the simulation.

The modular and changing nature of the MIRROR and MOSAR scenarios make the modeling and simulation a challenging task. Key challenges are the following:

- Complex structural variable systems consisting of modular satellite or array components. Modelica (and most other multi-body systems) do not directly support structural variable systems. The number of states must be constant during the simulation.
- Hybrid and stiff systems with discrete and continuous parts. The discrete elements result from the switching behavior of the SIs and algorithms for the control logic.
- Many disciplines involved: e.g. power and thermal management, robot control, and orbital mechanics
- Dynamic robot arm docking and pick & place operations with a high number of potential connection points. The term pick & place operation is used here for the following sequence of operations:
  1. The robot moves to the location of the module, ready to grasp the module
  2. The module is unfixed from the spacecraft and connected to the robot using the SIs
  3. The robot moves the module to the desired location
  4. The module is fixed at its new position via one or more SIs and disconnected from the robot SI.

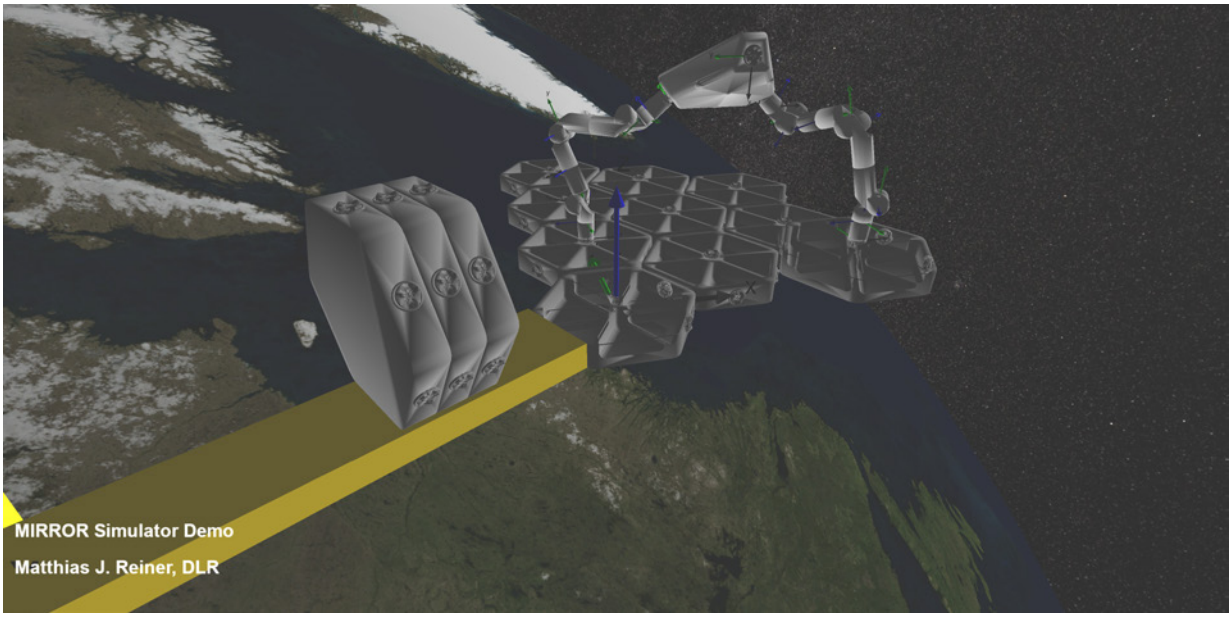

Figure 2. Visualization generated by the MIRROR Kinematic Dynamic & Graphical Simulator for an assembly scenario using the Multi-Arm Robot.

The model development and simulation of the scenarios with a focus on the Modelica aspects and unique design decisions will be described in the following sections. Since the models are quite complex only a brief overview can be given.

2 Model Development

The MOSAR Functional Engineering Simulator and the MIRROR Kinematic Dynamic & Graphical Simulator (both are abbreviated together as simulator, or individual as FES or KDG in the following) are built in the multi-physics modeling language Modelica to simulate the corresponding scenarios. The focus of the simulator is the walking robotic manipulator (WM and MAR) and the satellite platform with its modules and connectors (SIs, Hotdocks). The models were built up object-oriented, and allow to exchange individual components to generate different models for example to enable simulations with faster computational performance or more detailed models including additional dynamical effects.

The space environment of the MOSAR and MIRROR scenarios are implemented using the DLR SpaceSystems (Reiner and Bals 2014) and DLR Environment (Briese, Klöckner, and Reiner 2017) libraries, the robots working on these modules are modeled using the DLR Robot and RobotDynamics libraries (see (Reiner 2011) and (Tobias Bellmann, Seefried, and Thiele 2020)), the visualization is implemented directly in Modelica using the DLR Visualization library . A short summary of the content of these main libraries are listed in the following:

DLR SpaceSystems and Environment Libraries

The Modelica SpaceSystems library (SSL) has been developed at DLR over several years, to model space systems in a realistic space environment, ranging from satellites to launch vehicles, including their subsystems, components, and physical behavior, such as structural dynamics of solar panels and launcher stages. The SSL enables advanced controller design and verification, trajectory optimization, as well as development of path planning and other algorithms for new modular satellites, on-orbit servicing and reusable launcher concepts. It is a state-of-the-art enabling tool for future space dynamics and control activities. Various environmental effects on space systems can be represented with the co-developed environment models inside the DLR Environment library. In particular, an extendable and replaceable so-called World model is provided by the DLR Environment library. The World model defines coordinate systems, manages time and date, calculates sun and planet positions and provides state-of-the-art gravity models like EGM96. Optionally, models to calculate atmospheric conditions depending on the geodetic height of space vehicles, atmospheric drag, wind or other physical environmental effects can be activated within the analyses to provide a more realistic approximation of the relevant environmental conditions for space systems during all flight phases.

DLR Robots and RobotDynamics Libraries

The Modelica Robots and RobotDynamics libraries were designed to model serial kinematic robots. The libraries consist of components for the mechanical design of robots, including flexible elements and powertrains as

well as models for different robot control structures. They were developed and refined over many years and are used in various projects. The libraries focus on the efficient and exchangeable implementation of robot kinematics and dynamics. They also provide algorithms to solve forward and inverse kinematics problems. In addition, the libraries provide tools for the visualization of robots.

DLR Visualization Library

The DLR Visualization library provides an advanced, model-integrated visualization tool for Modelica models. The visualizer elements are directly part of the Modelica model using mechanical connectors and the visualization is generated directly at runtime. The library contains visualizers for basic shapes, CAD files, flexible bodies, surfaces, textures, light, energy and mass flow visualizers, analogue instruments and weather effects. A virtual camera system can be used to define the point of view manually or controlled by simulation. For space applications with a large difference in distances, a logarithmic Z-buffer has been implemented to be able to simulate the environment with an exact scale. SimVis is the software tool which displays the output generated by the Visualization Library.

Modeling Approach

A main simulation challenge for both MOSAR and MIRROR is the appropriate modeling of the assembly and reconfiguration of the modules. In both cases, a standard module is removed from its original position at the spacecraft, attached to an SI at a robot end effector, and integrated at a new position. Each of these transitions causes discontinuous changes in the mass and inertia properties of the involved subsystems which has to be adequately considered in the description of the dynamic system and selected modeling technique dealing with re-configurable systems.

The preferred solution for systems with a limited total number of configurations and with a limited number of configuration switches is the preparation of independent models for each configuration which are run sequentially. At a configuration switch from A to B, the model end state of configuration A will be used as the initial state of configuration B. In the MOSAR and MIRROR scenarios, both the number of possible configurations and the number of configuration switches exceeds a reasonable number for predefined model architectures. For a moderate number of configurations, the preferred method is working with controllable constraints. If the condition for a configuration switch is fulfilled, e.g. a body A has reached its final plug-in pose relative to body B, an event is triggered that causes a change in the connector constraint settings. In this example the constraint will switch from “free motion” to “rigidly connected”.

Mechanically, the constraint conditions are switching from zero forces/torques at plug and socket to identical position, velocity, and acceleration. The implementation is based on a method used at DLR-SR previously for rocket separation simulation (Acquatella and Reiner 2014) using force constraints with a variant of the Baumgarte stabilization. The basic idea is given in the simplified pseudo Modelica code in listing 1.

Listing 1. Simplified pseudo Modelica code to calculate switchable mechanical force constraint

```
// generalized position constraint
g_con = constraintForceAndTorque.frame_a.
    r_0 - constraintForceAndTorque.frame_b.
    r_0;
G_con = Frames.relativeRotation(
    constraintForceAndTorque.frame_a.R,
    constraintForceAndTorque.frame_b.R);

// generalized velocity constraint
g_con_dot = der(g_con);
G_con_dot = Frames.angularVelocity2(G_con);

// generalized acceleration constraint
g_con_ddot = der(g_con_dot);
G_con_ddot = der(G_con_dot);

// enable constraint switch
if not constrained then
 f_c = {0,0,0};
 tau_c = {0,0,0};
else
 g_con_ddot + 2*eta*g_con_dot + eta*eta*(
     g_con - pos_offset) = {0,0,0};
 G_con_ddot + 2*eta*G_con_dot
 + eta*eta*
 (Frames.Orientation.equalityConstraint(
 constraintForceAndTorque.frame_a.R,
     constraintForceAndTorque.frame_b.R) -
     angle_offset) = {0,0,0};
end if;
```

The code in listing 1 shows how the switchable connector is working: when the connector is open, the constraint force and torque are set to zero, so they can move freely. When the connector is closed, the force and torque between the two connector frames are computed such that the resulting relative position, velocity and acceleration are zero. The additional (Baumgarte) damping term (eta) is used to compensate numerical drift, caused by small integration errors. The parameter eta is used to define the stiffness and damping of the compensation.

However, extrapolating this technique to systems with a high number of switchable connectors would end up in huge stiff differential equation systems with an unacceptably high computational load. For a large number of potential configurations, a modeling technique was developed at the DLR during the MOSAR project that specifies only a general system architecture topology: the manipulator connected to the spacecraft with its end effectors using switchable constraints. Pick and place operations are then dynamically equivalent to adapting mass and inertia properties of the arrays to be assembled or the module stack to be reconfigured and of the modules at the robot end effectors.

The Tensor Body Concept

For this purpose, a new and innovative way to implement variable structure models was developed: individual configurations of the reflector or module stack are realized in the simulator by so-called tensor body models. These are, in essence, modifiable rigid bodies, which are consisting of individual sub-components and can change their inertia, mass and geometric shape. To clarify, since the word tensor is used very widely and differently in multiple fields, in this paper the term tensor is used to describe multidimensional arrays. These adaptable tensor bodies allow for activating and deactivating grid elements in terms of mass and inertia and also connectors depending on the actual operations. Thus, the simulator consists of a relatively simple model of the system dynamics and a logic component that adapts the tensor body model of the reflector or module stack at reconfiguration events as well as the corresponding tensor body located at the robot end effector. After each transition, the tensor bodies are re-computed based on the new configuration. This leads to an extreme reduction in the number of necessary states (depending on the number of modules) and improved numerical stability and computational speed. The properties of the components are dynamically re-calculated when the stack changes. The components of the body are defined via a tensor definition/syntax. They can also consider orientation and connectors of each individual component. Using the tensor body concept allows a simulation without restart/recompilation, because the number of states remains the same for the whole simulation duration. Reaction forces between the robot and platform/modules are fully considered. In addition, a special connection algorithm can check if an electrical connection can exist between the modules, considering the connectors of all involved components. It is implemented as a path finding search. The thermal balance is also computed in a similar way: a full grid of all possible module locations is dynamically updated when a reconfiguration occurs. This means the thermal capacity and thermal resistance of the connectors is updated depending on the current module type and the state of the connectors (including their current orientation). To make this possible, the thermal SI connectors are approximated by thermal resistors, which can switch from a very high resistance to a very low resistance value to simulate open or closed connectors. This results in a certain loss of accuracy, but leads to very fast simulation times and constant number of states for the thermal submodel of the tensor body. Since at an early mission state, most thermal data values are only rough estimates, this loss of accuracy can be accepted.

The current configuration of the modular satellite or mirror array is defined by the variable called shapeTensor with three dimensions for each spatial direction (tensor_nx, tensor_ny and tensor_nz). This maximum tensor size cannot be changed without a re-compilation of the Modelica model, however the content of shapeTensor can be modified during the simulation to account for the reconfiguration of modules.

Different types of modules have been implemented, they are given a number to differentiate them, while 0 stands for no module at that tensor location. The properties of these individual module types can be changed via the corresponding parameters (e.g. mass, inertia, thermal capacity, SI configuration etc.). The initial configuration of the arrangement of these modules is defined by a parameter shapeTensorInit.

Whenever the robot moves a module from A to B, the tensor body is re-calculated automatically (by triggering a recomputeTensor event) internally and the required connectors are automatically opened and closed as needed to realize the new configuration. In addition, tensor body elements are also located at each SI of the robot, such that the handling of all types of modules is possible. If no component is connected to a robot SI, the tensor body at the SI is just a dummy mass with a very small inertia and mass (called epsMass) to avoid a division by zero and to enable a constant number of states.

To be able to identify which exact module is located at each position within the tensor grid, each existing component is also given a unique ID in addition to its general module type number.

Each individual module within the tensor also has an individual orientation (relative to the tensor body base frame), and up to seven connectors for the hexagonal elements (and six for the cubic modules).

The following simplified pseudo Modelica code listing 2 shows the base algorithm for the tensor body, where some details were omitted for brevity.

Listing 2. Simplified pseudo Modelica code for the main algorithm of the tensor body

```
when {initial(),edge(recomputeTensor)} then
 //check possible connections with path
     finding algorithm
 (poweredTensor,connectionTensor,...) :=
 Components.Electrical.
     VarTensorBodyConnectionAlgorithm(..)
 //initialize mechanical properties
 m := 0;
 r_CM := {0,0,0};
 //iterate over spatial tensor dimensions
 for ii_x in 1:tensor_nx loop
  for ii_y in 1:tensor_ny loop
   for ii_z in 1:tensor_nz loop
   // check if element exists
   if shapeTensor[ii_x, ii_y, ii_z] > 0
       then
    //add center of mass (not normalized
       yet)
    //massPerEle is a vector which contains
        the individual mass of the
       specific module type
    //r_ele is a tensor with relative
       position vectors for each possible
       element location in the 3D tensor
       grid.
    r_CM := r_CM + massPerEle[shapeTensor[
```

```
        ii_x, ii_y, ii_z]]*r_ele[ii_x, ii_y
        , ii_z, :];
    //add total mass
    m := m + massPerEle[shapeTensor[ii_x,
        ii_y, ii_z]];
   end if;
   end for;
  end for;
end for;
//divide by total mass, avoid div/0
if m < epsMass then
 m := epsMass;
end if;
r_CM := r_CM/m;
//compute inertia tensor w.r.t. previously
     computed CM
//init
I := zeros(3, 3);
for ii_x in 1:tensor_nx loop
 for ii_y in 1:tensor_ny loop
  for ii_z in 1:tensor_nz loop
  //inertia solid cuboid with Steiner term
  // check if element exists
   if shapeTensor[ii_x, ii_y, ii_z] > 0
       then
    // current rotation for the element
    R_ele := rotMatTensor[ii_x, ii_y, ii_z
        , :, :];
    //inertia for specific element
    I_ele := inertiaPerEle[shapeTensor[
        ii_x, ii_y, ii_z],:,:];
    I := I + R_ele*I_ele*transpose(R_ele);
    // helper vars -> distance to CM
    rx := r_ele[ii_x, ii_y, ii_z, 1] -
        r_CM[1];
    ry := r_ele[ii_x, ii_y, ii_z, 2] -
        r_CM[2];
    rz := r_ele[ii_x, ii_y, ii_z, 3] -
        r_CM[3];
    // Steiner tensor
    I := I + massPerEle[shapeTensor[ii_x,
        ii_y, ii_z]]*[ry^2 + rz^2,-rx*ry,-
        rx*rz; -rx*ry,rz^2 + rx^2,-ry*rz;
        -rx*rz,-ry*rz,rx^2 + ry^2];
   end if;
  end for;
 end for;
end for;
//avoid div/0
if I[1, 1] + I[2, 2] + I[3, 3] < epsMass
    then
 I[1, 1] := epsMass;
 I[2, 2] := epsMass;
 I[3, 3] := epsMass;
end if;
 //update of heat grid model and power
     state of SI connections omitted
(...)
end when;
```

The code listing 2 shows the main mechanical algorithm to re-compute a tensor body when it is changed. A logical algorithm in the model (not shown) triggers the recomputeTensor event and checks which tensor bodies are involved and how they should be changed. For a pick & place operation this leads to the re-computation of the properties of the main tensor body, as well as the corresponding tensor body at an end effector on one of the robot arms. For each involved tensor body, first the new Center of Mass (CM) is computed based on the change of the stack of modules or array elements represented in the tensor variable shapeTensor, which is modified by the logical algorithm. Using the newly computed CM as the new center, an updated inertia tensor is computed for the tensor body which considers all individual orientations of the individual tensor body elements as well as the individual inertia tensors for each module. Since after a pick & place operation, no module can be left at an end effector of a robot, a very small (dummy) mass and inertia are used to approximate this. Since all involved tensor bodies are updated and recalculated with the same event at the same time, no discontinuities can occur, and the total mass always stays constant.

Modeling Overview

The base coordinate system for the KDG and FES simulation is an Earth Centered Inertial (ECI) system. The orbit of the robot and satellite or antenna array can be defined by using appropriate start position and velocity vectors, together with the simulation date. The date is used to compute the current rotation of the earth and the position of the moon and sun. Other planets are neglected because only the LEO (Low Earth Orbit) or GEO (Geosynchronous Equatorial Orbit) are considered here. This allows for a very generic simulation of nearly every LEO and GEO orbit and can also be used to consider lighting and heating conditions from the resulting sun position. For the simulation it is considered, that the servicer spacecraft and client satellite or platform are already docked and that the possible locations of all modules are known beforehand. For the cubic shape of the modules this is quite straight forward and leads to three-dimensional tensor shape, for the hexagonal modules of the MIRROR project, the possible locations are stored in a table (called r_ele in listing 2) which is used within the tensor body calculation.

The spacecraft modules (SM): The spacecraft modules are modeled as tensor bodies. It is generally a full SM stack with individually and dynamically activatable and de-activatable SM for implementing the modular reconfiguration, depending in the SIs' respective lock status. The SMs can have individual inertia properties, depending on their respective internal equipment. Each cubic module has up to 6 connectors (around all the sides). These can be used to grasp the modules, and to walk the robot over them. For the hexagonal modules a similar approach was implemented in the KDG simulator where each module can have up to seven SI connectors along the sides of the hexagonal modules and one on top. The simplified models of the electric and thermal domain are integrated in the tensor models.

The robot models consist of kinematic and dynamic models. For MOSAR they also include powertrain and sensor models. For the detailed model variants also the

joint torque and position controllers are implemented very close to their actual implementation using the DLR Robot-Dynamics and Modelica Standard library. The robots are equipped with SI interfaces to enable walking and module grasping capabilities. The SI modules are modeled as switchable force/torque constraints, as described previously. The robot model includes the joint motor and friction as well as joint nonlinear elasticity for the complex simulator variants. For the simulation the full dynamic coupling of the robot's interaction with the satellite is considered.

The simulators contain many additional features and aspects which can only be briefly mentioned and referenced here. The following list gives an overview over the main features of the simulator:

- Dynamic satellite platform and modules based on the tensor body approach: After each transition, the tensor bodies are re-computed depending on the new configuration. This allows simulation without restart/recompilation for the complete simulation scenario (number of states remains constant). Reaction forces between the robot and platform are fully considered and the heat flow between components can be modeled. A path finding connection algorithm can check if an electrical connection to a power source module exists.
- Visualization of components and important properties.
- Detailed WM and MAR robot model using data from Unified Robot Description Format (URDF). For the WM also powertrain models with motor friction brake, nonlinear flexible joint models for gearbox and nonlinear friction as well as joint position and torque controllers are included.
- Variable module design with individual mass and inertia as well as individual SM configuration including heat and power properties.
- Detailed orbit model: orbital dynamics, including sun and moon.
- Simplified surface heat model including solar, albedo, deep space and planet infrared radiation. The models consider the position of the sun, moon and earth for the shadows as well as the spacecraft attitude. The implementation is based on (Posielek 2018) but is extended to be compatible with the tensor body concept.
- Surface contact model with friction: Contact forces between robot TCPs, modules and platform are considered and are dynamically updated depending on the shape of the tensor body. The implementation is based on previous work at DLR-SR, see (Reiser 2021).
- To stabilize the platforms and satellites, when the robots are moving, a Quaternion attitude controller is implemented, as a simplified Guidance, navigation and control (GNC) system.
- To allow the simulator to obtain signals from other software components and high-level control systems a UDP interface was implemented. This allows for the communication with an external high-level WM controller (developed by the Institute of Robotics and Mechatronics DLR-RM). The implementation is based on (Thiele et al. 2017).

3 Demonstrator Overview

Since the MIRROR project is still ongoing, this section will focus on the end results for the MOSAR project incorporating the FES. The FES is only one small part of the overall demonstrator setup of the MOSAR project, which involved many project partners: SpaceApps, GMV, MAG-SOAR, Thales Alenia Space (France and UK), SITAEL, Elidiss Technologies, University of Strathclyde, Glasgow and DLR (SR and RM) (see (Letier, Yang, et al. 2019)). Figure 3 shows pictures of some results of the MOSAR project.

The demonstrator setup consists of a physical assembly of prototype modules and the WM as well as multiple software components. The WM was designed to be able to work under the earth's gravity conditions. For the much larger MAR of the MIRROR project a weight compensation construction is being developed for the demonstrator on earth.

For the final demonstration of the MOSAR project multiple scenarios were developed and executed. For each scenario the starting and desired end configurations of the module stacks and the WM pose were defined with the help of the design tool, which was based on the FES, and a static simulation analysis of the configurations, to check the module SI connector compatibility and power source configuration.

A planning algorithm (developed by GMV) uses this configuration information to plan a sequence of operations for the WM and the SI connectors to move and reorient the modules collision free from the starting to desired end configuration. Because of the limited reach of the WM, this can also include relocating the WM between different steps of the operational plan.

The planning module communicates with a high-level robot controller developed by DLR-RM. This controller can communicate with the FES as well as the real hardware setup, since the interfaces were defined analogously. To ensure the safe operation of the WM and to validate the operation plan. The complete sequence is simulated using the FES first, before moving the real hardware.

The high-level WM controller can move the robot both in position mode (Cartesian and joint command possible), as well as in a compliance mode, using the robots joint torque sensors. For the simulation the high-level WM

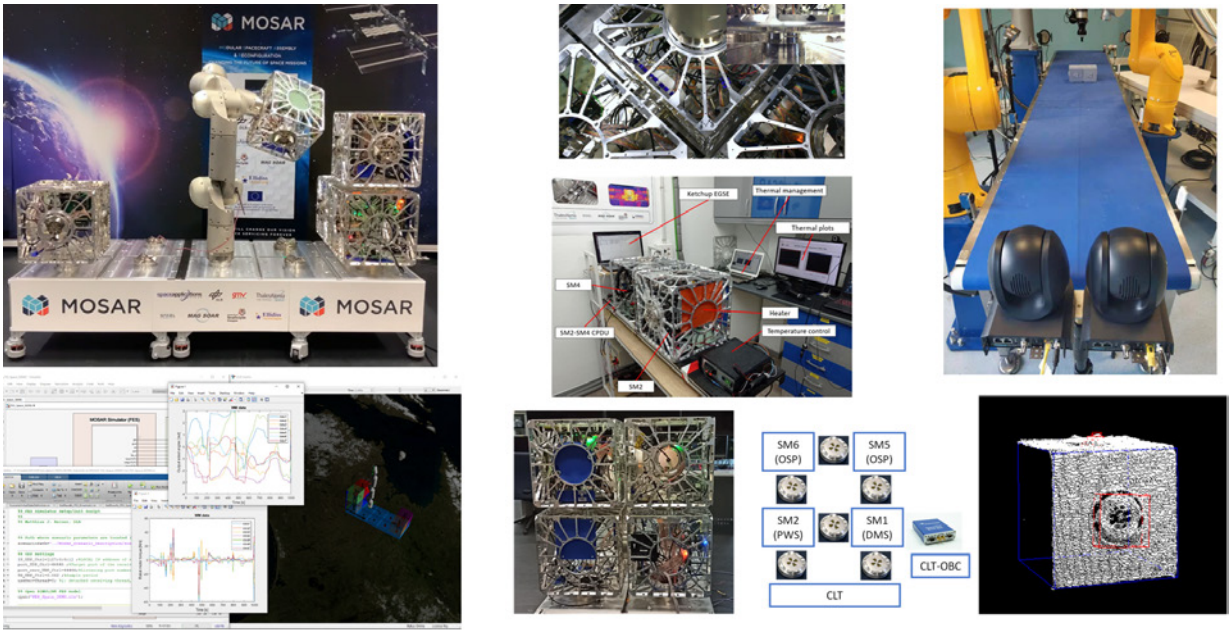

Figure 3. The mosaic overview taken from an internal MOSAR report shows some results obtained within the MOSAR project from the involved project partners SpaceApps, GMV, MAGSOAR, Thales Alenia Space (France and UK), SITAEL, Elidiss Technologies, University of Strathclyde, Glasgow and DLR (SR and RM). The top left shows the physical demonstrator setup with the WM relocating an SM. The top middle gives a closer look on the SIs and a thermal testing setup. The top right shows a setup for the visual inspection of the modules. The lower left shows a screenshot of the MATLAB interface of the FES with plotted simulation results. In the lower middle a closer view of an SM configuration and its corresponding software setup can be seen. The lower right corner shows the result of a camera-based damage inspection of an SM.

controller sends the desired joint position or torque commands as well as SI commands to the FES using the UDP interface. The low-level joint axis controllers for both the torque and position control mode are implemented within the FES. The FES uses this input data to compute the resulting motion and sends simulated sensor information back to the WM controller. The communication is synchronized by using a blocking UDP implementation, which waits for the corresponding resulting data package before executing the next step. The high-level WM controller also communicates with the planning module, to enable a re-planning in case something goes wrong or a plan is not feasible and needs to be re-planned.

After a completed simulation run, the resulting simulation data generated by the FES can be analyzed by an expert team before starting the same procedure with the real hardware setup to ensure a safe operation. For example, joint limitations and safety distances to objects can be checked there.

Since most project partners did not use Modelica directly, the Modelica model used within the FES was exported as an S-function, using a tool provided by Dassault Systems Dymola. This S-function can then be used and configured in MATLAB/Simulink. The visualization can also work together with the generated S-functions.

Since the definition of the scenarios and parameters is quite complex, the parameter settings, as well as the output processing were done using MATLAB scripts (.m files) which also enables the use of algorithms to simplify the definition and output generation.

4 Conclusions and outlook

The simulation of the complex scenarios in Modelica with the FES was possible because of the newly developed features, such as the tensor body concept for the structural variable SM stack, as well as the integration of many previously developed Modelica libraries at DLR-SR.

Using the complete demonstrator setup and all developed software components from all project partners, all demonstration scenarios could be successfully performed at the end of the MOSAR project.

The concept was used in the MIRROR project which enabled a very fast development time for the simulator there, and with small adaptations could also be used for many similar applications both on-orbit as well as on earth (e.g. building construction).

However, the tensor body concept and switchable force constraints also lead to a significant increase in the overall model complexity (especially the required logical parts) and some approximations and simplifications are necessary and restrict what is possible to simulate.

An extension of the Modelica language (and simulation tools) to directly handle structural variable systems or new developments within similar languages such as the modeling language Modia (Elmqvist et al. 2021) could improve

this aspect in the future and would enable the direct implementation of structural variable systems in a more classical object-oriented way.

Acknowledgements

The work presented here would not have been possible without the great support of colleges at the DLR-SR who provided Modelica libraries that were part of the simulation as well as the good collaboration within the MOSAR and MIRROR project with the involved partners.

The MOSAR part of this work was funded by the European Commission Horizon 2020 Space Strategic Research Clusters on Space Robotics and Electric Propulsion programme under grant number 821966 (MOSAR: Modular Spacecraft Assembly and Reconfiguration). The following companies and institutions were involved in the MOSAR project: SpaceApps, GMV, MAGSOAR, Thales Alenia Space (France and UK), SITAEL, Elidiss Technologies, University of Strathclyde, Glasgow and DLR (SR and RM).

The MIRROR part of this work is still ongoing and funded by the European Space Agency (ESA) in the framework of the Technology Research Program (contract No. 4000132220/20/NL/RA) entitled Multi-arm Installation Robot for Readying ORUS and Reflectors (MIRROR). The following companies and institutions were involved in the MIRROR project: SpaceApps, Thales Alenia Space, LKE, Leonardo, Frentech Aerospace and DLR (SR and RM).

References

Acquatella, P. and M. Reiner (2014). "Modelica Stage Separation Dynamics Modeling for End-to-End Launch Vehicle Trajectory Simulations". In: *Proceedings of the 10th International Modelica Conference*, pp. 589–598.

Bellmann, T. (2009). "Interactive Simulations and advanced Visualization with Modelica". In: *Proceedings of the 7th Modelica Conference*, pp. 541–550. URL: https://www.modelica.org/publications.

Bellmann, Tobias, Andreas Seefried, and Bernhard Thiele (2020). "The DLR Robots library - Using replaceable packages to simulate various serial robots". In: *Asian Modelica Conference 2020*. Linköping Electronic Conference Proceedings 174. Linköping University Press, pp. 153–161. URL: https://elib.dlr.de/138327/.

Briese, L., A. Klöckner, and M. Reiner (2017). "The DLR Environment Library for Multi-Disciplinary Aerospace Applications". In: *12th International Modelica Conference*. DOI: 10.3384/ecp17132929.

Deremetz, Mathieu, Gerhard Grunwald, et al. (2021-12). "Concept of Operations and Preliminary Design of a Modular Multi-Arm Robot using Standard Interconnects for On-Orbit Large Assembly". In: *72nd International Astronautical Congress (IAC),* URL: https://elib.dlr.de/147153/.

Deremetz, Mathieu, Pierre Letier, et al. (2020-10). "MOSAR-WM: A relocatable robotic arm demonstrator for future on-orbit applications". In: *International Astronautical Congress, IAC 2020*. IAF. URL: https://elib.dlr.de/139962/.

Elmqvist, Hilding et al. (2021-09). "Modia - Equation Based Modeling and Domain Specific Algorithms". In: *14th International Modelica Conference*. Ed. by Martin Sjölund et al. Linköping Electronic Conference Proceedings 181. Linköping University Electronic Press, pp. 73–86. URL: https://elib.dlr.de/144872/.

Hellerer, Matthias, Tobias Bellmann, and Florian Schlegel (2014-03). "The DLR Visualization Library - Recent development and applications". In: *The 10th International Modelica Conference 2014*. Linköping Electronic Conference Proceedings. LiU Electronic Press, pp. 899–911. URL: https://elib.dlr.de/92153/.

Kümper, Sebastian, Matthias Hellerer, and Tobias Bellmann (2021-09). "DLR Visualization 2 Library - Real-Time Graphical Environments for Virtual Commissioning". In: *14th Modelica Conference*. Ed. by Martin Sjölund et al. Modelica Association and Linköping University Electronic Press, pp. 197–204. URL: https://elib.dlr.de/144780/.

Letier, Pierre, Torsten Siedel, et al. (2020-10). "HOTDOCK: Design and Validation of a New Generation of Standard Robotic Interface for On-Orbit Servicing". In: *International Astronautical Congress, IAC 2020*. IAF. URL: https://elib.dlr.de/139963/.

Letier, Pierre, Xiu Yang, et al. (2019-05). "MOSAR: Modular Spacecraft Assembly and Reconfiguration Demonstrator". In: *15th Symposium on Advanced Space Technologies in Robotics and Automation*. URL: https://elib.dlr.de/129392/.

Posielek, Tobias (2018). "A Modelica Library for Spacecraft Thermal Analysis". In: *The American Modelica Conference 2018*. URL: https://elib.dlr.de/123229/.

Reiner, M. (2011). "Modellierung und Steuerung eines strukturelastischen Roboters". PhD thesis. Technische Universität München.

Reiner, M. and J. Bals (2014). "Nonlinear inverse models for the control of satellites with flexible structures". In: *Proceedings of the 10th International Modelica Conference*, pp. 577–587.

Reiser, Robert (2021-09). "Object Manipulation and Assembly in Modelica". In: *14th International Modelica Conference*. Ed. by Martin Sjölund et al. Modelica Association and Linköping University Electronic Press, pp. 433–441. URL: https://elib.dlr.de/144864/.

Thiele, Bernhard et al. (2017-05). "Towards a Standard-Conform, Platform-Generic and Feature-Rich Modelica Device Drivers Library". In: *12th International Modelica Conference*. Ed. by Jiri Kofranek and Francesco Casella. Linköping University Electronic Press, 2017, pp. 713–723. URL: https://elib.dlr.de/118601/.