MMX Rover Simulation - Robotic Simulations for Phobos Operations

F Buse and A Pignède and J Bertrand and S Goulet and S Lagabarre

The MMX Rover, developed by CNES and DLR, will fly to and explore the surface of the Martian Moon Phobos within the JAXA Martian Moon Exploration Mission. It will be the first wheeled locomotion system in a milli-g environment. In the development of the rover, simulations have been used to test and develop its robotic activities. This paper presents the multi-physics simulations that are being used. The overall simulator setup and its main components are discussed. To provide appropriate simulations for the various topics while maintaining a unified simulator, a modular approach was required. The different modules and their role will be outlined. For this, Dymola's implementation of the Modelica modeling language provides the basis, especially regarding multi-body dynamics, and the possibility to include external libraries, e. g. for environment interaction, control logic and visualization. Finally, examples for the simulator used in driving, uprighting, alignment and separation will be presented. These examples illustrate the approach on experiment design, setup and result evaluation. To date the MMX Rover simulator is regarded as an indispensable development and analysis tools, especially since representative lab experiments are much limited when designing a robotic system for milli-g operations. It is also planned to be used during operations phase for planning and analysis.

F. Buse, A. Pignède, J. Bertrand, S. Goulet and S. Lagabarre, "MMX Rover Simulation - Robotic Simulations for Phobos Operations," 2022 IEEE Aerospace Conference (AERO), 2022, pp. 1-14, doi: 10.1109/AERO53065.2022.9843391.

MMX Rover Simulation - Robotic Simulations for Phobos Operations

Fabian Buse
German Aerospace Center (DLR)
Münchener Str. 20
82234 Wessling, Germany
fabian.buse@dlr.de

Antoine Pignède
German Aerospace Center (DLR)
Münchener Str. 20
82234 Wessling, Germany
antoine.pignede@dlr.de

Jean Bertrand
Centre national d'études spatiales (CNES)
18 Av. Edouard Belin
31401 Toulouse Cedex 9, France
jean.bertrand@cnes.fr

Sébastien Goulet
CS Group - France
22 Av. Galilee
92350 Le Plessis-Robinson, France
sebastien.goulet@cnes.fr

Sandra Lagabarre
Centre national d'études spatiales (CNES)
18 Av. Edouard Belin
31400 Toulouse, France
sandra.lagabarre@cnes.fr

***Abstract*—The MMX Rover, developed by CNES and DLR, will fly to and explore the surface of the Martin Moon Phobos within the JAXA Martian Moon Exploration Mission. It will be the first wheeled locomotion system in a milli-g environment. In the development of the rover, simulations have been used to test and develop its robotic activities.**

This paper presents the multi-physics simulations that are being used. The overall simulator setup and its main components are discussed. To provide appropriate simulations for the various topics while maintaining a unified simulator, a modular approach was required. The different modules and their role will be outlined. For this, Dymola's implementation of the Modelica modeling language provides the basis, especially regarding multi-body dynamics, and the possibility to include external libraries, e. g. for environment interaction, control logic and visualization.

Finally, examples for the simulator use in driving, uprighting, alignment and separation will be presented. These examples illustrate the approach on experiment design, setup and result evaluation. To date the MMX Rover simulator is regarded as an indispensable development and analysis tools, especially since representative lab experiments are much limited when designing a robotic system for milli-g operations. It is also planned to be used during operations phase for planning and analysis.

TABLE OF CONTENTS

1. INTRODUCTION

The Martian Moons eXploration (MMX) mission of the Japan Aerospace Exploration Agency (JAXA) will explore the Martian Moons Phobos and Deimos in the second half of the current decade. One of the objectives of this ambitious mission is to return a sample from the larger of the two Moons, Phobos. To support this, the French National Center for Space Studies (CNES) and the German Aerospace Center (DLR) are developing a small rover that will land on Phobos [1]. Its tasks are to provide on surface measurements that can also be used as contextual information to the samples as well as to gather vital data aiding in the spacecraft landing [2]. This rover will also be the first wheeled system in a milli-g environment [3], [4].

One of the critical parts of the rover is its locomotion subsystem, consisting of four individually driven wheels attached to four individually actuated legs [5]. The legs allow the rover to be put into a stowed configuration during transit. Once on Phobos the rover can then deploy its legs. This locomotion system must fulfill three main tasks: driving, uprighting and alignment. The first activity executed by the rover on Phobos will be using the locomotion system to upright the rover. This means, once the rover has come to rest on the surface, it needs to unfold and refold its legs to reorient itself onto its belly and then stand up. The second task of the locomotion system is the alignment of the rover body. This will be used to point the rover towards the Sun for optimal battery charging as well as to point cameras or scientific instruments. A system called "SKA" (which stands for "le système für die Kontrolle of the attitude") will calculate the required rover orientation changes based on its current configuration and measurements from the rover's sun sensor.

Due to the low gravity on Phobos, lab experiments of all these activities are limited and of limited representativeness. Further, the surface conditions on Phobos have large uncertainties, especially on the scale of the rover. Thus, simulations were required from early on and will continue throughout the whole mission, including the operations and post-mission phase.

Landing and motion on a small planetary body has not been done often, e. g. by the hoppers of the Hayabusa 2 mission, and *wheeled* locomotion in such low gravity has only been envisioned once [6], but finally not realized because of budget constraints.

978-1-6654-3760-8/22/$31.00 ©2022 IEEE

Simulations of planetary exploration rovers have been done numerous times already, sometimes using special dedicated tools: e. g. Artemis [7] or ROSTDyn [8], sometimes implemented within commercial simulation frameworks such as MATLAB/Simulink[2] or Simpack[3]. Likewise simulative investigation of effects and mobility in milli-g is not new [9]. Yet the subject discussed here is new in terms of the combination of tools, the applications and the importance it has attained for the project and success of the mission.

We chose a hybrid approach for the tool selection. The implementation of the Modelica[4] modeling language in the Dymola[5] simulation framework and IDE builds the basis. On top of it are various in-house developed libraries for visualization or contact dynamics. A broad description of this approach was first published in "Rover Simulation Toolkit" (RST) [10]. The present publication discusses more details of this tool chain as well as its applications within the MMX rover mission.

The usage of the simulations can be broadly separated into two categories: First, simulations that need to be executed once or a few times only to help with a specific issue. These simulations usually try to answer a well-defined question, along the lines of: "How does the rotational velocity of the joints unfolding the solar panels affect rover stability?" To answer these kinds of questions, a single known parameter, the opening velocity in this case, has to be varied. A limited and known number of variables are the metrics used to answer the question.

The second type of questions, the more difficult ones, are much more open and are along the lines: "Does the current design of the uprighting algorithm function as desired, what is its reliability and where does it fail?" For these cases the parameters that need to be varied and the number of evaluated metrics is much larger, and often at least partly unknown. For these cases Monte Carlo style simulations are necessary.

2. MMX ROVER SIMULATION MODEL

This section describes the general design and setup of the simulation software and its models. First, the setup and usage of the software tools is described, then the various aspects like design of the physical rover model, contact and environment simulation are discussed. The simulations focus on the the dynamic behavior of the rover's mechanical system interacting with the environment. Other domains like thermal, electrical or energy are only modeled were strictly required.

Simulator Setup

To be able to solve both cases without the need of maintaining different simulator environments a unified structure is needed, see Figure 1 for an outline of our implementation. In this structure the "Simulator Core" is a stand alone executable. It is developed in Dymola and mostly based on the modeling language Modelica. The model of the "Rover system" is a multi-domain model. It focuses on the rover structure, mechanisms and sensors. By adding "Control Logic", "Contact Models" as well as an "Environment", simulations of the rover are possible. The "Environment Generator" enables parametric descriptions and procedural generation of the environment. The environment generator, parts of the contact models as well as the control logic are not suitable to be implemented in Modelica. They are integrated into the "Simulator Core" via the Modelica C interface. With this unified structure it is necessary to be able to configure the simulator appropriately for the different use cases, and thus allow an optimal balance between accuracy and computational complexity. This is achieved by using the modularity of Modelica and providing variants of the different components of the simulator with various levels of detail. To make sure that all of these variants are based on the same parameters, and thus the same rover version, the model structure and implementation is separated from the rover parameters.

With this setup, the "Simulator Core" can be configured pre-runtime to fit the needs to answer specific questions. The user configures the simulator core to fit the scenario's needs and manually selects parameters.

To efficiently solve the second use case of more complex simulations the "Simulator Core" is embedded into an framework implemented in Python[6]. This framework allows to deploy a large number of simulation instances and takes care of parameter assignment and result gathering. The experiment is designed by assigning distributions to the various parameters and initial conditions as well as selecting the desired rover control logic. Once started each instance is configured from a single parameter file. The result file generated by each instance contains selected outputs as well as the parameter file contents. Further, the result file can directly be used as basis for a second simulation of the same situation, either enabling a more detailed analysis by activating the 3 D visualization or comparing different control algorithms.

Rover System Model

The rover system model implemented in Modelica describes all physical components of the rover, see the most central element of Figure 1. This includes all dynamic systems such as locomotion, shutters or the solar array. Internal components of the rover, such as the on board computer or the battery, are not modeled independently but are combined into a single rigid body.

Model Structure—The rover model structure closely matches the physical structure of the rover. Each model and sub-model shown in Figure 2 can be independently configured or is existing with varying levels of detail. The encapsulation into sub-models allows the implementation of various variants and levels of detail for each sub-model without affecting other components. Moreover, not every sub-model is required for all simulations. For example the "Mechanical and Electrical Chassis Support System" (MECSS) is solely relevant for separation simulations and can thus be left out for uprighting, driving and many other applications. Further, implementation and parameters are separated, to ensure consistency among the different variants. All parameters are combined into a separate tree that mirrors the structure of the model implementations.

Mechanisms and Actuator Models— All active elements, marked blue in Figure 2, of the rover contain at least one kind of actuation device. If only models with the highest level of detail are used, the complexity of the total rover simulation

[2] https://www.mathworks.com/
[3] https://www.3ds.com/products-services/simulia/products/simpack/
[4] https://modelica.org/
[5] https://www.3ds.com/products-services/catia/products/dymola/
[6] https://www.python.org/

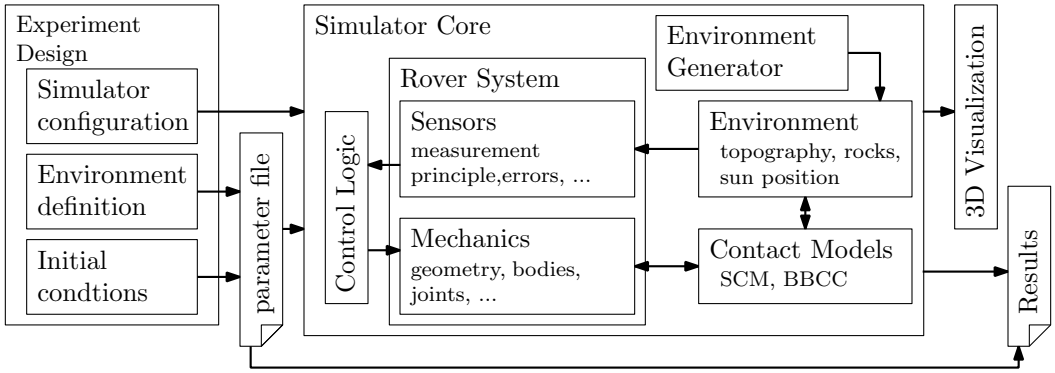


Figure 1. Overview of the general structure of the MMX rover simulator. Outlining the software structure of the simulation software used in the MMX Rover mission.

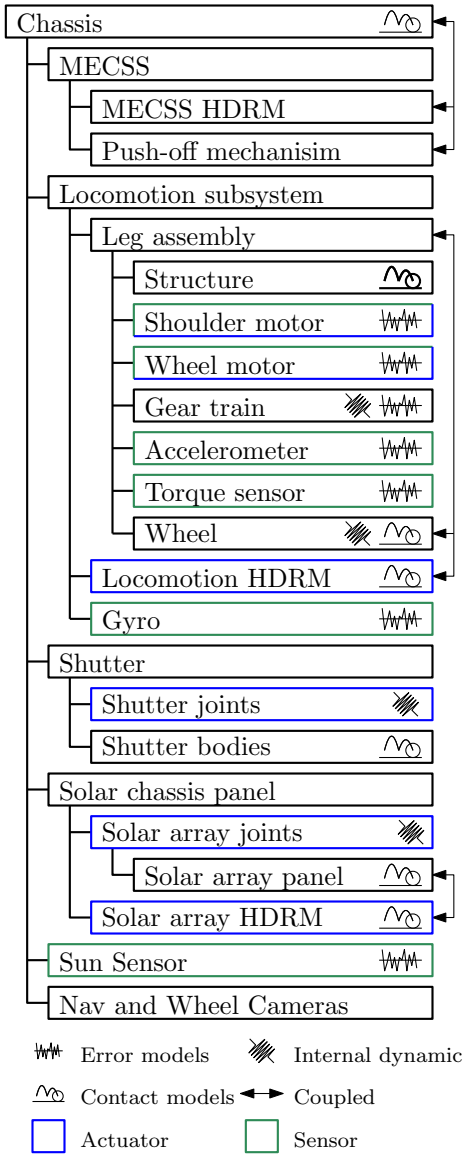


Figure 2. Model structure of the physical rover model displayed as a tree. Pictograms show model characteristics of the different sub-models.

model increases drastically, but this is hardly needed. Usually at least three variants are available: a rigid version that completely locks this joint, a simplified version and a detailed version. This enables a precise pre runtime configuration of the simulator for different applications.

Good examples for the use of rigid variant actuators are the solar panel joints. As their mechanisms are based on shape memory alloys, and once opened are locked in place, there is no benefit of simulating them during driving operations. They are further good examples of detailed versions, as when analyzing the impact of the opening sequence on the rover stability, the precise behavior is of interest. In this case actuators, based on a lookup table derived from physical experiments, are used. Together with these lookup tables, the "Hold Down and Release Mechanisms" (HDRM) for the solar arrays, that are disregarded in other applications, are added to the simulation model.

Similar to the solar panel joints, detailed models of the shutter joints are required to analyze their dynamic effects when opening. As theses joints are constructed from a combination of counteracting rotational springs, matching 1 D drive train models were implemented.

In case of the locomotion subsystem, a well-defined simplified version can be used in most applications. Since the joints are position-controlled and the external torques are small, the assumption of an ideal drive usually applies. This assumption is valid as long as this approximation complies correctly with all acceleration and velocity limits.

Sensor Models—For the sensors, a similar approach as for the mechanisms was selected. The signals of the modeled sensors, marked in green in Figure 2, are used by the on board software, or the "Control Logic" in simulation, to govern the rover's behavior. Thus, these sensors and their signals must be modeled to correctly match in type and quality with their real counterparts.

In total the rover simulation model provides the following sensor signals, these are identical to the sensor signals of the real rover:

- The relative direction to the Sun provided by a Sun sensor on the rover top panel
- Torque sensors at the shoulder joints
- Angular positions at the shoulder joints provided by a set of two potentiometers per shoulder and a Hall effect sensor

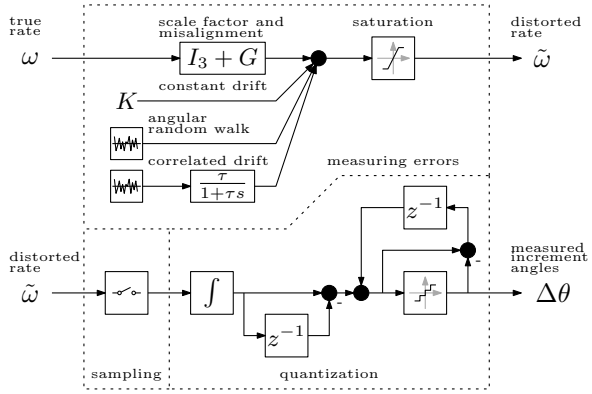


Figure 3. The gyroscope signal processing model. Separated into signals, measuring errors, sampling and quantization.

- Angular positions of the wheels provided by Hall effect sensors
- Tri-axial acceleration sensors at the shoulder joints
- A two-axis gyroscope for the orientation in the rover's body

Again, models of each sensor can vary in their level of detail. The simplest versions only apply limits and offsets. Whereas more detailed versions also add parasitic effects of the measurement principle as well as noise and drift.

The various noise signals are generated by the Modelica Advanced Noise[7] library. As it is seed-based, it generates reproducible pseudo-random noise signals.

A good example for a detailed sensor model is the gyroscope, its implementation follows the theory explained in [11], [12]. In addition to the true signal derived from the simulation, different factors like finite precision and accuracy or quantization must be considered. Figure 3 shows the implementation of the gyro model. First, the true rates are combined with a constant gyro drift and limited white noise that is processed to get the correlated gyro drift and angular random walk. Next, saturation is applied to the signal. At this stage, distorted angular rates representing the continuous signal with errors are available. By applying sampling and quantization this rate is converted into the quantized increment angles around each of its sensitive axes.

Environment Interaction

A key factor in the analysis of most robotic actions performed by the rover, is to accurately capture its interaction with the environment. This means foremost interaction of the rover with soft deformable regolith, loose and embedded rocks or bedrock. All sub-models marked with the contact model pictogram in Figure 2 provide both means of contact described in this section. Contact detection and force calculations are provided by the internally developed and currently mostly unpublished "Contact Dynamics Library". This library is tailored towards providing contact models for the simulation of off-road vehicles. The models used in this context are a penalty based general purpose contact model called "Body Body Contact written in C" (BBCC) as well as a specialized soft soil contact model called "Soil Contact Model" (SCM).

[7] https://github.com/DLR-SR/AdvancedNoise

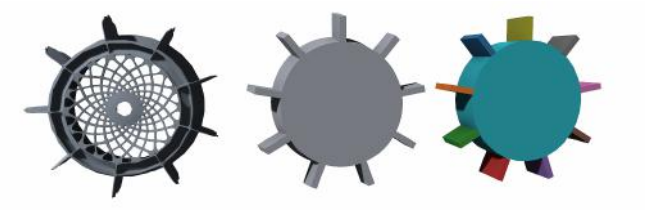


Figure 4. Rover wheel representation for contact. Original MMX wheel CAD (left), SCM simplification to a mesh of the main traction features (center), BBCC simplification where each color represents an own contact element (right).

The BBCC model uses a collision detection based on the publicly available libccd[8] library. Because the collision detection is restricted to convex shapes only, more complex shapes, like the rover's wheels, need to be separated into smaller "subshapes". The simplifications done to the rover's wheel are shown in Figure 4. The reaction forces and torques are then calculated based on a simple Hertzian model for the normal force and a Coulomb friction model for the tangential forces. Applications of this contact model is the contact between rocks and parts of the rover as wells as contact between parts of the rover system as used when simulating the separation from the spacecraft.

Even though simpler contact models are available for regolith contact within the Contact Dynamics Library, only SCM provides contact between arbitrarily shaped objects and regolith with still an adequate performance. The only other terramechanical models that are in general capable fulfilling this requirement, are DEM-based and computationally expensive. SCM uses a node based 2.5 D representation of the surface, where each of the equidistantly spaced nodes represents a vertical column of soil. By modifying a node's height and internal states the soil can be deformed. The bearing capacity of each node is then used to calculate the resulting forces. The objects in contact with the soil are described as meshes.

Contact shapes for SCM are simplified as well, as very high resolution meshes are computationally expensive and do not provide any additional precision once their resolution is significantly higher than the surfaces resolution, see Figure 4. For more details on SCM see [13].

Whether and which contact model is activated for the various rover components depends on the respective analysis and its requirements. For example in a simulation with an already fully deployed rover, where the focus lies only on driving in soft soil, only SCM is enabled for the wheels. Whereas in a simulation regarding uprighting both contact models, SCM and BBCC, are enabled for the chassis, solar array, shutters, legs and wheels. Of course some simplifications are often applicable, e. g. when the solar panels are not unfolded during the simulation, they may be combined into a single large element. For later analysis, the forces acting onto each contact element as well as the binary information if a contact was present, are logged at each time stamp.

Environment Generation

The environment in simulation has four main components:

1. The evolution of the Sun's position over time.

[8] https://github.com/danfis/libccd

2. The gravitational slope and magnitude, i. e. the average angle between gravity and the terrain on a large scale.
3. The terrain topography limited to an area inside 100 m radius, everything above is depicted in the gravitational slope
4. The distribution of rocks; this, in reality also being part of the topography, needs to be separated due to the drastically different interaction between the rover and a discrete rock or regolith.

The simplest of the three aspects is the gravity. Since this is just defined by the gravity vector direction and magnitude and is constant during a single simulation, it's simply a parameter.

The Sun's position can either be set to a fixed point over a simulation, this is usually used if it has no immediate impact on the current simulation. Alternatively the Sun can follow predefined trajectories, based on lookup tables generated from ephemeris for specific locations and dates on Phobos [14][9].

Much more complex are the generation of terrains and rock distributions usable for simulation. This is handled by a custom library specifically developed for this purpose. The general process is based on a procedural terrain i. e. a terrain that is created by a deterministic parameterized algorithm. Further rocks adhereing to a desired size distribution are then placed on the previously generated terrain.

Many publications in the field of computer graphics use Perlin noise to generate terrains [15], [16], but even though these terrains often look similar to the eye, it is not possible to reproduce the slope vs. distance distributions observed for different planetary bodies with this method. The Perlin noise algorithm superpositions a given number of noise octaves, where the noise amplitudes decrease with increasing frequency. The ratio is defined by a factor called lacunarity λ. This process leads to a fractal, self identical result when comparing the terrain to itself on different scales.

To enable the surface generator to generate terrains that follow a desired slope vs. distance distribution, the idea of the Perlin noise algorithm was adapted to use a function to scale its amplitude. This function must be available in the form $v(L)$ where its output is the RMS (root mean square) height deviation and the input is the corresponding baseline L. In this implementation the baseline of an octave, the inverse of its frequency, is defined as the distance in space where on average the noise function went from its minimum to its maximum. The implementation uses a two dimensional gradient coherent noise function $\mathcal{GCN}(u_1, u_2)$ as a base. To generate the terrain height at a given position $(x,\ y)$, the resulting amplitude is the sum over all octaves:

$$h(x,y) = \sum_{k=1}^{n_{\text{Octaves}}} v(f_{\text{min}}^{-\lambda k})\, \frac{\mathcal{GCN}(x f_{\text{min}}^{\lambda k}, y f_{\text{min}}^{\lambda k})}{2} \tag{1}$$

The additional factor $1/2$ is required as the output of the gradient coherent noise function $\mathcal{GCN}$ is within $[-1, 1]$ and must be scaled to have a deviation of 1 between its minimum and maximum value. The additional parameters lacunarity λ, lowest applied frequency f_{min}, and number of octaves n_{Octaves} must be selected. The lacunarity $\lambda = 2$ was selected for the environment generation of Phobos. With a given lacunarity λ, the lowest frequency and number of octaves can be derived from the desired smallest L_{min} and largest L_{max} distances that are to be reproduced correctly by the algorithm.

$$f_{\text{min}} = {}^{1.0}/_{L_{\text{max}}} \tag{2}$$
$$f_{\text{max}} = {}^{1.0}/_{L_{\text{min}}} \tag{3}$$
$$n_{\text{Octaves}} = \lceil \log_\lambda ({}^{f_{\text{max}}}/_{f_{\text{min}}}) \rceil \tag{4}$$

With this adaption, it is possible to generate terrains that follow an arbitrary distribution that can be defined by describing the RMS height deviation as a function of a baseline.

The function used to describe the RMS height deviation v as a function of a given baseline L is:

$$v(L) = v_{L0} \left(\frac{L}{L_0} \right)^H \tag{5}$$

The parameters roughness gain v_{L0}, reference baseline L_0 and the Hurst exponent H describe the topography, see [17] for more details. L_0 is defined to be 1 in this implementation. This makes it possible to generate terrains applicable for Phobos that adhere to the distributions agreed within the mission in the Environment Requirement Document (ERD) [18]. To validate the implemented method, the parameters that are used to define the terrain are identified based on the generated surface. This identification process uses randomly selected pairs of points of which the distance and height deviation is calculated. Equation 5 is then fitted to match the data. Figure 5 shows a comparison of the desired distribution vs. the results of generated terrains. Slight deviations between different instances, especially at larger baselines, are to be expected as the generated surface area is finite and thus large baselines are less common. When averaging over multiple terrains, a good match of the desired and the generated parameters can be observed.

The rock distribution, similar to the terrain generation, uses a predefined distribution to calculate the number of rocks expected on a given terrain in a certain size range. Any function providing the number of rocks $n_{\text{rocks}}(d)$ larger than

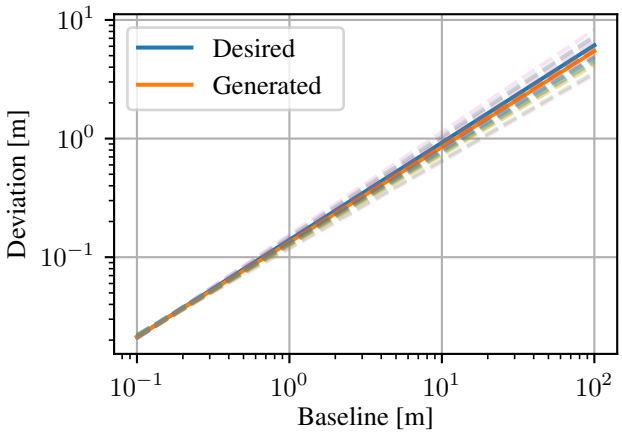


Figure 5. Results of the topography generation. Displayed are the results of twenty generated terrains (dashed) with different seeds. Each individual terrain slightly deviates, whereas the average (orange) over multiple terrains matches the desired (blue) distribution.

[9] https://ssd.jpl.nasa.gov/

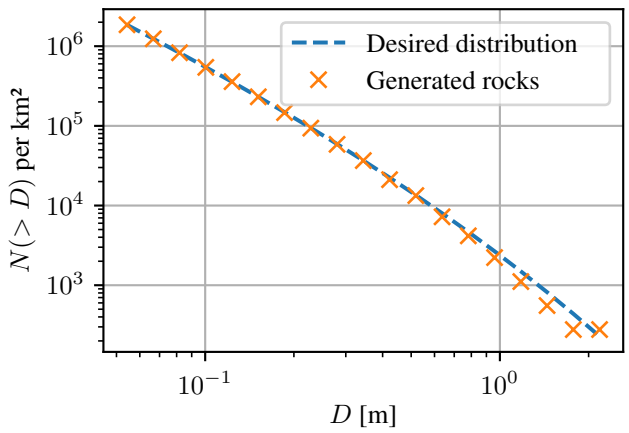


Figure 6. Results of the rock distribution generation, comparing the results when binning the generated rocks by size and calculating their occurrence per unit area

a defined size d over a unit area can be used. In this application the following function

$$n_{\text{rocks}}(d) = n_{D0} \left(\frac{d}{D_0} \right)^{\kappa(d)} \tag{6}$$

$$\kappa(d) = \alpha + \beta \left(\frac{d}{D_0} \right)^{\gamma} \tag{7}$$

with n_{D0} as well as α, β and γ being the parameters describing the rock distribution, has been used. D_0 is a predefined reference rock size.

Theoretically, based on this equation and a given area, the rock distribution could be generated by iteratively decreasing d from some very large start value until another rock needs to be generated, then repeat this until a user-defined minimum rock size $d_{\min}$ has been reached. Even though possible, this is computationally inefficient. By generating rock size bins, that are geometrically spaced in between the largest rock in the expected area $d_{\max}$ and the user defined minimum rock size $d_{\min}$ and then linearly interpolating in between, the distribution can be generated more efficiently. With the number of rocks larger at the upper bound $n^+ = n(d^+)$ and at the lower bound $n^- = n(d^-)$ within a bin, the total number or rocks in a bin can be calculated. By then generating rocks with a uniform size distribution within that bin, a linear interpolation is achieved. By selecting the number of rock size bins, the quality of the interpolation can be adjusted. See Figure 6 for a comparison against the expected rock distribution used in the rover missions agreed for in the ERD [18]. The generated rock distribution shows a good match over most of the range. At the upper end of the distribution the absolute number of rocks per unit gets small, thus rounding errors and the errors due to interpolation increase.

After the rock distribution has been generated, the actual rocks and their position on the terrain must be generated. The rocks themselves are then generated by deforming a sphere represented as an icosahedron. First, the sphere is deformed along the x, y and z axes to match a specific sphericity and overall size. Then each vertex radius is scaled based on a Voronoi noise with its spherical coordinates as input. By

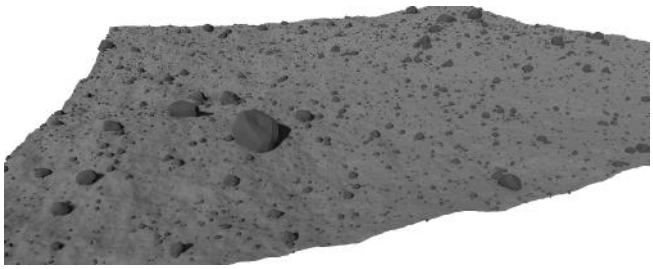


Figure 7. Generated Phobos environment with a size of 60 m by 60 m

deforming and modifying the scale and bias of the Voronoi noise, the roundness of the rock can be adjusted. Both sphericity and roundness are randomly picked for each rock based on predefined ranges. Finally the generated list of rocks are randomly placed on the terrain. This is done by applying a uniform distribution to the x and y coordinates and using the surface height of the terrain at this x and y position as the z coordinate basis for the rock. How much each rock is buried, is again randomly selected based on a predefined range. Figure 7 shows a generated environment based on the parameters used in Figure 5 and Figure 6.

Since neither the terrain nor the rock distribution can generate a "real" landing site on Phobos, the simulator can only provide a statistically correct and correctly looking version. All analysis must take this into account by not relying on a single terrain instance, but instead use a large number of variations.

Rover Control

To perform any activity in simulation the correct command sequence need to be issued to all active components of the rover. To do so, all actuators are modelled with a command interface corresponding to the real counterparts. This means for example that the HDRMs of the shutters are triggered by a Boolean whereas the joints require a target position and a velocity limit as their model includes the motor control algorithm.

The complexity generating the required combination of inputs depends on the simulated activities. If a single actuation, like opening a shutter, is in focus, it is trivial to directly define those. The robotic activities uprighting and Sun pointing both rely on some higher level logic that governs the rover's actions. To realistically evaluate the performance of these functions, understanding the effects resulting from the logic interaction with the environment is key. Explicitly defining all required inputs for those more complex actions, the direct approach, is not viable anymore. Instead the logic needs to be implemented in a suitable matter and the kinematic control algorithms translating commands on chassis level to joint level are required.

The kinematic calculations from the control modes available in the locomotion flight software, i. e. driving, alignment and uprighting, see [19] for more details, are translated to Modelica and thus directly available. For the future it is planned to use the actual flight software of the locomotion system in the simulator.

To provide means to build the high level logic, the ModelicaLua library [20] is integrated into the simulator. This

library enables to use Lua, a scripting language commonly used in professional applications and game development, in Modelica. The interfaces to the rover are abstracted into a separate Lua library, allowing for a simple and intuitive way to script the rover behavior. This library provides an interface to read all available sensor signals as well as command the rover on a similar level as done within the flight software. For example a script that drives the rover for $15\,\mathrm{s}$ with a velocity of $0.01\,\mathrm{m\,s^{-1}}$ and then lowers the rover body down to $20\,\mathrm{cm}$, looks like this:

```
-- create instance of the rover
local Rover = MMXRover_Library.new()
-- configure rover to be in drive mode
Rover:enableKinematicCommands()
Rover:enableSlipCorrection()
-- wait until simulation time
-- is exactly at 10s
Modelica:wait_until(10)
-- start rover drive
Rover:drive(0.01,0)
-- wait 15s
Modelica:wait(15)
-- stop rover drive
Rover:drive(0.0,0)
-- wait 5s
Modelica:wait(5)
-- lower rover height to 20cm
Rover:align({0,0,1},0.2)

-- wait until rover is at target
while not Rover:legsAtTarget()
do
  Modelica:wait(1)
end
```

With this abstraction of the rover interface and third party libraries available for Lua, especially tools like vector math[10] and state machines[11], quick and comprehensive implementation of the required functions is possible.

Visualization

A big part in modeling and simulation is result interpretation. The default numerical outputs are often not enough to fully understand a situation. This is especially true, once interactions with the environment, deformable surfaces and moveable rocks are part of the simulated scenario. Thus the DLR Visualization 2 library [21] to provide a detailed 3 D representation of the simulation, see Figure 8, is extremely valuable.

By introducing virtual cameras, the loop to the DLR navigation experiment [22] can be closed for representative testing prior to flight.

3. APPLICATIONS

Driving

Introduction—To assist the development of the locomotion subsystem, see [5], different simulations in the theme of driving were conducted. These simulations range from answering simple aspects like the maximum safe rover acceleration to more complex questions like an analysis of the rover's point turn performance in various conditions. The latter is presented as an example application in more detail in this subsection.

[10] https://github.com/bjornbytes/maf
[11] https://github.com/kyleconroy/lua-state-machine

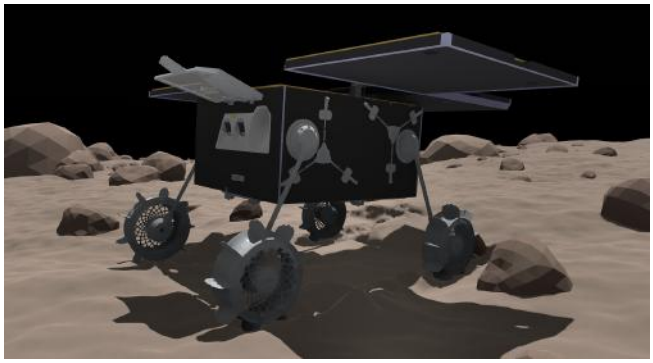


Figure 8. Visualization of the MMX Rover simulation

As the rover has no steered wheels, it has to rely on differential steering to navigate. A series of point turn maneuvers in varying environments is simulated, to outline its capabilities. Goal of this analysis is to identify how sensitive the action is to regolith parameters and topography.

Methodology— The rover model for driving simulation is configured to focus on the interaction of the locomotion subsystem with the ground through its wheels. Thus, all aspects that are not related to driving, such as shutters and solar panels, are modeled as simply as possible. Motor models in the locomotion subsystem are configured to adhere to any set acceleration and velocity limits, but do not model any internal dynamics. Flexibilities in the gears or in the legs themselves are not required to be modeled, as the maximum forces and torques experienced still lead to negligible deflections.

The rover is initialized fully deployed on a surface and commanded to turn the wheels such that in ideal conditions a $90°$ point turn would result. The ratio between the realized and commanded yaw angle is used as a metric to evaluate the rover's ability to turn.

Two sets of analysis are performed. First the metric is compared on two terrains over a grid of the regolith parameters friction angle and cohesion to identify the broad behavior. The friction angle is varied in steps of $5°$ between $15°$ and $40°$, the cohesion is varied in steps of $10\,\mathrm{Pa}$ between $0\,\mathrm{Pa}$ and $50\,\mathrm{Pa}$. It has to be noted, that these values cannot be directly related to any physical soil parameters for Phobos. A completely flat terrain is used as a baseline to compare against a terrain with a natural surface, generated with the tools discussed above. Then a statistical analysis is done with a single set of regolith parameters over a larger number of terrain topographies.

Results— Figure 9 shows the results of both experiments. When comparing the natural and the flat terrain, an expected decrease in performance due to uneven terrain can be observed. The uneven surface leads to uneven and unfavorable weight distribution, i. e. some wheels have a higher sinkage in the regolith. This results in higher resistance at these wheels with higher loads, that cannot be compensated by an increase in traction. Further, it can be observed that the change in cohesion has a much larger effect than a change in regolith friction angle. This is due to the behavior of the regolith, a

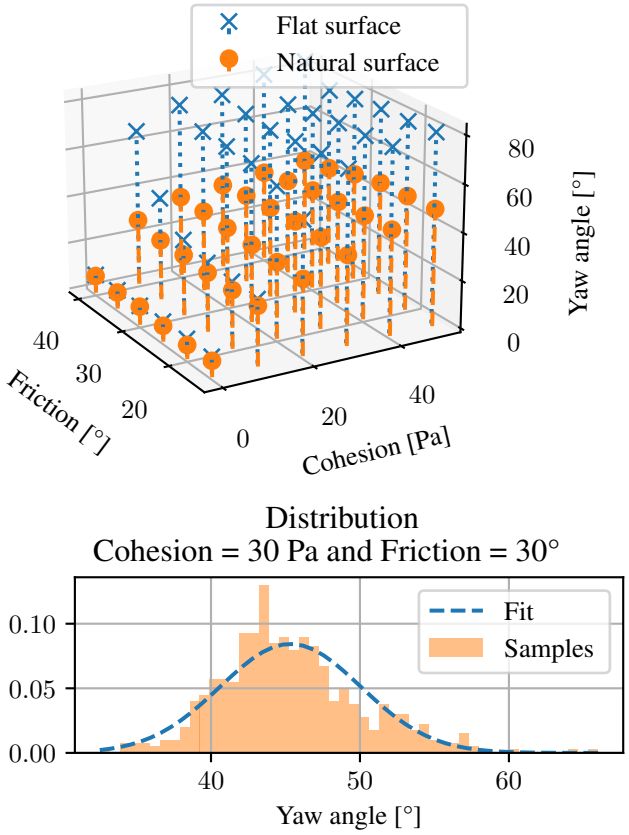


Figure 9. Heading angle change performance function of soil cohesion and friction angle, comparison between flat and sloped surface geometry

lower cohesion also leads to higher sinkages and thus higher resistances.

The result of the second experiment, shown in the lower plot in Figure 9, is a distribution over different instances of similarly parameterized terrains. The histogram matches a normal distribution with an average at $45.3°$ and a standard deviation of $4.8°$. The minimum heading angle change recorded over the 600 samples is $32.5°$, the maximum is $66.0°$.

Two conclusions are drawn from these results. First, on very soft regolith point turns are not efficient, a heading angle change should be realized by elongated curves when possible. Second, depending on the local topography, even in good conditions, the actual achieved yaw angle may vary by a large margin.

Separation

Introduction— Before the rover reaches Phobos, it will be separated from the spacecraft about $50\,\mathrm{m}$ over ground. A device called "Mechanical and Electrical Chassis Support System" (MECSS) interfaces between spacecraft and rover. It provides a mechanical connection, power and communication during the cruise phase. The mechanical connection is ensured by four "Hold Down and Release Mechanisms" (HDRM) at each corner. A spring based push-off mechanism in the center allows for a defined separation on release.

While tests on Earth can ensure the functionality and reliability of the individual parts, analysis of the rover 3 D trajectory is done in simulation. Goal of this simulation is to identify the sensitivity of the process to the parameters of the different components like rover mass, properties of the push-off spring and effects of the HDRM opening. Further, a confirmation that the current configuration can fulfill all requirements is desired.

Two requirements are defined by mission analysts:

- The separation velocity along the push-off direction must be $20\,\mathrm{cm\,s^{-1}}$ with a $3\,\sigma$ allowed uncertainty of $2\,\mathrm{cm\,s^{-1}}$.
- The rover's rotation rate must be $0\,°\,\mathrm{s^{-1}}$ with a $3\,\sigma$ allowed uncertainty of $10\,°\,\mathrm{s^{-1}}$.

Methodology—For this application a detailed simulation of the whole rover is not required. Rather, the rover behaves mechanically as an inert mass until the uprighting sequence starts on the surface on Phobos. Thus, the rover model for separation simulations is simplified to a single body with the respective mass and inertia. The spacecraft is modeled similarly as a single body. This "passive rover" has two interfaces to the MECSS: the contact between the push-off plate and the rover's belly and the last HDRM unopened. As the four HDRMs are opened one by one and the pauses are sufficiently large, no effects of the first three are required to be modeled. The push-off plate is connected to the spacecraft via a spring while the plate itself can interact with the rover chassis by contact. The contact is modeled with the BBCC contact model and simplified to five contact points one on each corner of the X shaped plate and a single point in the center.

During the HDRM opening a single screw is released and retracted into the rover chassis. As this imparts some momentum on the rover, a model of the mechanism was first analyzed in more detail. Due to the high mass ratio between the bolt and the rover's body this model could be simplified in the final simulation to directly impart the resulting force on the chassis without explicit modeling of the dynamics.

Analysis of the separation was done in two stages. In the first stage only one parameter was varied at a time. This permitted to already eliminate parameters that don't have an impact on the separation. On the other hand some parameters were identified to have a significant impact. The whole simulation was then designed in a Monte Carlo style, with random variations of all parameters deemed interesting:

- rover mass properties (mass, inertia, center of gravity)
- push-off spring stiffness
- impulse imparted by HDRM
- component alignment
- spacecraft motion during separation

A total of 100 000 samples are simulated and the state (position, angle, velocities etc.) $1\,\mathrm{s}$ after the separation is recorded.

Results— As a representative example of the Monte Carlo analysis the effect of the spring constant on the separation velocity and the tumbling of the rover is given in Figure 10. These special histograms are to be read as follows: On the x axis is the spring constant, it was separated into 100 bins of equal range because the parameter was varied uniformly, the red line marks the nominal value. On the y axis is the velocity, translational along the line of separation (rover height) or rotational about the rover side (direction of the

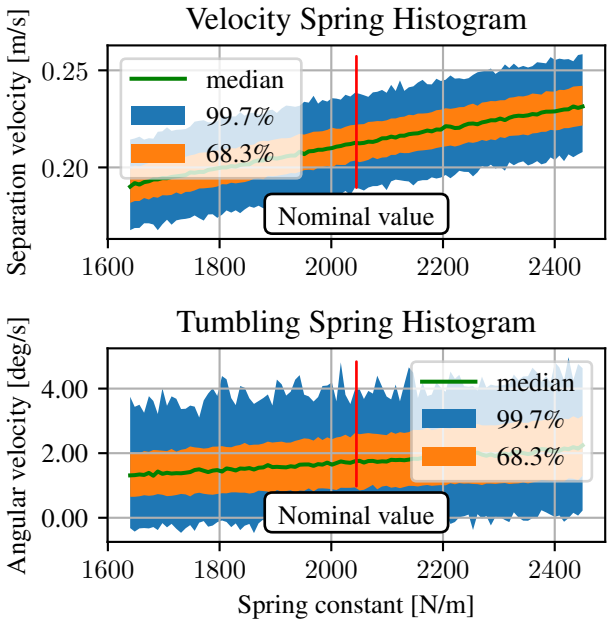


Figure 10. Rover translational velocity along the line of separation and rotational velocity about the sideways axis 1 s after separation as 2 D histogram of the push-off spring constant

wheels) respectively. To show a statistical evaluation three regions are shown, the green line is the median velocity inside each spring constant bin, in orange and in blue are the $1\,\sigma$ and $3\,\sigma$ regions of each bin, outliers are not shown.

The top plot of Figure 10 shows as expected that the velocity increases with a higher spring constant, the function can be approximated as linear and the spread is constant. It is important to note that the median separation velocity value for a nominal spring constant is higher than the required $20\,\mathrm{cm\,s^{-1}}$, the $2\,\mathrm{cm\,s^{-1}}$ uncertainty requirement is only met by the $1\,\sigma$ region.

The bottom plot of Figure 10 is easy to interpret qualitatively, a higher spring constant introduces more energy, thus more tumbling, again with a quasi linear relationship. Here too, important results are visible: Because the spring is not exactly aligned with the middle of the rover bottom plate, it is a few mm closer to the back, the median of the rotation motion about the rover side axis is significantly away from zero. Still this is not critical as even the $3\,\sigma$ region in Figure 10 always is under the requirement of $10\,^\circ\,\mathrm{s^{-1}}$ margin. It must be noted however, that this is only the motion about one axis. The tumbling about the other two axes is smaller, but the total tumbling magnitude comes very close to the requirement when the spring constant attains higher values.

Uprighting

Introduction—Uprighting is one of the most critical phases of the MMX rover mission. It describes the phase between the rover coming to rest on the surface after descent and the rover standing on its legs ready to begin its mission. In this phase the rover must reorient itself autonomously from any side towards its belly and then safely stand up. An example of this sequence is shown in Figure 11

This phase has changed during the development of the rover. The initial design foresaw an absolute orientation knowledge based on a fiber optic gyro. To aid with recovering the rover when it is resting on its left or right side, an additional set of mechanisms was added to the rear of the rover. This mechanism was named “flaps”, each flap could unfold separately and push the rover towards its belly. As the development evolved, the fiber optic gyro was removed and finally replaced with a more weight and power efficient two-axis MEMS gyroscope. Simulations showed that without without any absolute orientation information available, the benefit of the flaps mechanism does not exist. Thus in its current configuration, the flaps have been removed. In all these decisions, the different versions of the uprighting algorithm had to be developed and tested in simulation. Goal of the uprighting simulation in general is to capture the impact of different environmental aspects as well as the rover design and the algorithm on the success rate.

The analysis performed to compare the uprighting performance with and without flaps will be discussed in more detail. An algorithm that does not use any orientation information is used in this analysis. Thus, it has to be based on a universal sequence of unfolding and refolding the legs and flaps in a predefined order. As in a universal sequence both the flaps, positioned at the rover rear, and the legs, positioned on the rover’s left and right sides, have a chance of being actuated while the rover lies on their respective mounting side, an additional chance of failure must be considered. This additional failure results from the assumption that the probability of blockage increases significantly when either the legs or the flaps are driven through regolith. It was agreed to treat any case in which a leg or flap is dragged through the regolith as a failure case in this analysis. Goal of the analysis is to identify how this impacts the overall success rate.

Methodology—The general setup of uprighting simulations is straight forward. The rover is initialized on a randomly generated terrain. Once the rover is stationary, the uprighting sequence is initiated. After the uprighting sequence has been completed or a timeout has occurred, the final rover state is used to determine success. In addition to the final rover state, other metrics, like the ratio between contacts with rocks and regolith, are recorded for later evaluation. The uprighting algorithm is implemented as a state machine in Lua. Environment contact is enabled on all rover subsystems. When sensors like the gyro are used, the most detailed sensor models are selected. As the terrain and rock distributions as well as the actuator and sensor errors are all seed-dependent, it is possible to filter for interesting cases based on results and re-run them with enabled visualization to reconstruct and identify the failure mechanism.

In the simulations performed to analyze and verify the uprighting algorithm’s performance, parameters either are varied for each simulation run in a larger set or changed between sets. Parameters that are varied per run are:

- terrain and rock distribution seed
- the direction of gravity, within the agreed range with respect to the average terrain normal
- the rover’s initial position and orientation
- the rover’s initial translational and rotational velocity
- the magnitude and direction of realization errors of the locomotion system
- if applicable, the magnitude and direction of orientation

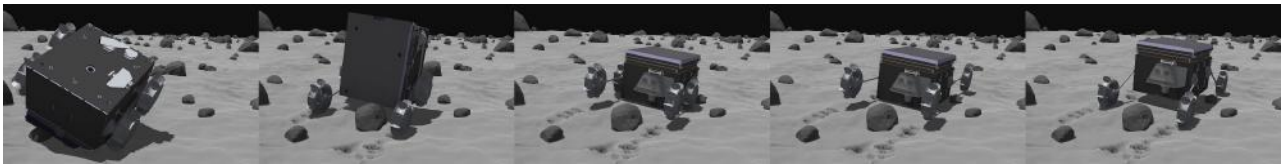


Figure 11. Uprighting sequence, showing the rover orienting itself from its top to its belly and standing up

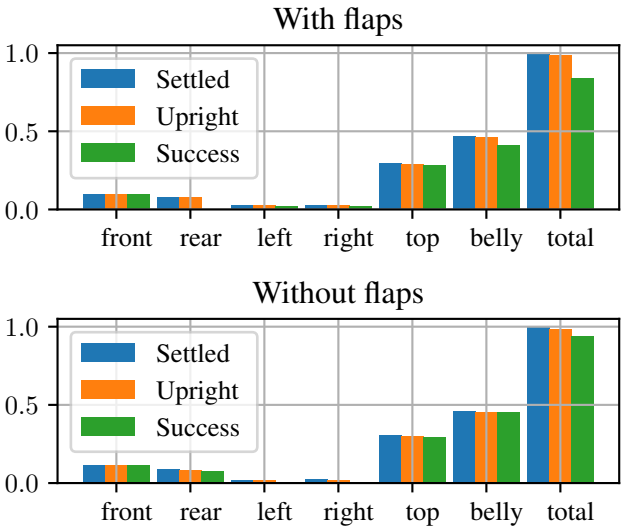


Figure 12. Uprighting simulation results, comparing the success and upright rates of a rover with and without flaps

knowledge error

Parameters that are varied between sets are:

- terrain and rock distribution parameters
- gravity slope range
- limits for locomotion realization errors
- limits for gyro measurement errors
- regolith parameters

In case of the comparison used to determine the impact of flap removal a single set with the parameters deemed most likely is used. Results for the two rover versions are then compared.

Results—An intuitive approach to analyze the different effects of removing the flaps, is comparing the success rate of uprighting when binning the data based on the settled orientation of the rover. The settled orientation is described by the bottom side of the rover after it came to rest before uprighting starts. For every case two indicators are identified. First, "upright" is true if the rover is oriented correctly at the end of uprighting without regards to any other factor. The "success" criterion is true, if the rover is "upright" and has not performed any actions deemed dangerous. Actions that are classified as dangerous in this analysis, are actuation of legs or flaps while the rover lies on their respective mounting sides.

When comparing the results in Figure 12, it can be observed that the total chance of upright does not decrease when removing the flaps. This would have been the case if the rover often got "stuck" on its left or right side without the use of flaps. Further, the total chance of success increases as the number of dangerous actions performed in total is decreased. This is because the rover orientation when settled, is not distributed uniformly. The left and right sides of the rover, due to the irregular shape as well as the position of the center of gravity, are less likely than the rear side.

In summary the simulation shows that uprighting success is not higher with flaps with the current sensor suite. This was one argument in descoping the flaps system. The decision was taken by the rover lead engineers, considering also that removal of a single purpose system saves mass and decreases complexity.

Sun Pointing and Alignment Simulations

Introduction—Both for optimal battery charging as well as for the operation of the scientific instruments the rover is required to change its body orientation and height. With the configuration of the rover's locomotion subsystem this is possible by actuation of all legs and wheels. As this maneuver is performed in a natural environment, the interaction with the surface and with rocks must be considered. The most critical of these alignment maneuvers is the first one performed just after uprighting to optimally recharge the rover's battery. This maneuver has to be executed autonomously.

The system within the rover that is responsible to generate the required commands to move the rover, is called "Le *S*ystème für die *K*ontrolle of the *A*ttitude" (SKA).

The SKA system uses the current configuration of the locomotion subsystem in combination with the measurements taken by the Sun sensor to determine an orientation change to point the solar array towards a fixed direction that ensures the battery charging over the next hours, while keeping the rover in a stable position and trying to avoid contact between instruments or solar panels and the ground. As for uprighting, the removal of the fiber optic gyroscope led to a significant simplification of the associated algorithm although leading to a sub-optimal pointing on some areas: the rover now points towards the Sun direction around noon in Phobos local time through a sequence composed of successive alignment iterations, while keeping the height of the rover within a pre-defined safe range, see Figure 14 for an illustration of this algorithm. Several iterations are performed in order to compensate for the alignment realization error resulting from the complex interaction with the regolith. Figure 13 shows this sequence when operating nominally. Risks to the rover are tipping over, ground collisions or excessive burying of the wheels.

As the analysis of this action requires consideration of the environment, the performance of the locomotion system as well

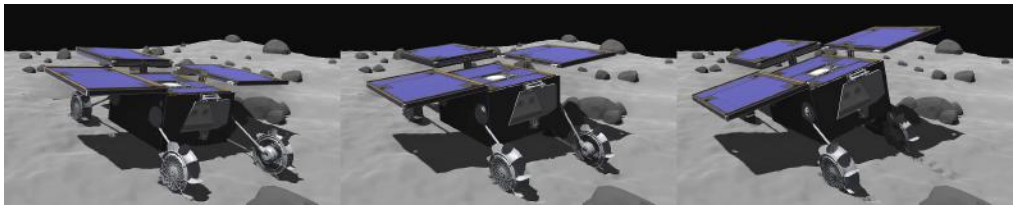


Figure 13. Example of an alignment sequence as commanded by SKA. The three images, from left to right, show the step by step alignment of the rover towards the Sun.

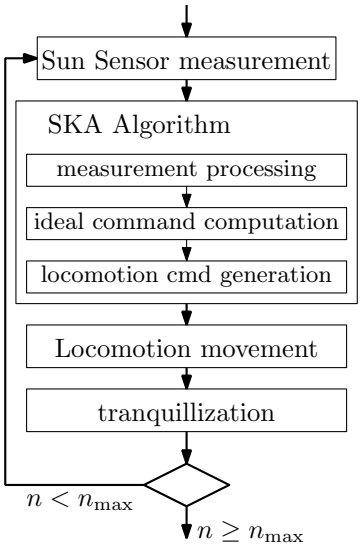


Figure 14. Flow of the SKA algorithm

as the algorithm, simulations are required. The evaluation of the SKA performance, mainly the typical energy expectancy over a Phobos day, will help in describing the possible areas where the rover would survive as function of the landing date.

Methodology—Very similar to the simulations performed for uprighting, the logic of SKA is implemented in Lua and integrated into the simulator.

The simulation is configured to start just after uprighting. Thus the rover has successfully uprighted itself, is in a stable position on its legs and has deployed its solar array. Contact is enabled for the wheels, chassis as well as solar array as these components could come into contact with either rocks or the regolith. Sun trajectories considered in these simulations are randomly selected from a pre-computed set and correspond to potential landing sites and dates.

An important factor in this process are the errors in both the Sun sensor and the locomotion system. Errors in the Sun sensor are modeled as white noise. For the locomotion system three error sources are identified and modeled. The error in alignment of the shoulder axis is modeled as a uniform error. A rotational offset in the shoulder constant over a full run is added, this is consistent with uncompensated offsets in the potentiometers measuring the shoulders angles. Finally, backlash is introduced in the locomotion's gear train.

For the analysis of the initial alignment of the rover, multiple metrics are of interest. First, whether the rover does not tip over or get in similarly dangerous situation during the pointing activity. This is captured in the rover's stability margin, describing how much the angle to the gravity vector could deviate in any direction without the rover tipping over. Second, whether the rover does manage to align itself sufficiently well with the optimal direction to provide sufficient charging. This is captured with the Sun declination after pointing, describing the angle between the solar array normal and a vector pointing towards the Sun. Finally, whether the clearance between the rover's belly or solar panels and the ground or rocks is sufficient. This metric describes the shortest distance between any point on the rover and the ground, along a direction normal to the plane spanned by the rover wheels. The ground clearance is separately measured between the rover and regolith as well as the rover and rocks. The final value than is the minimum of both. As the it is possible that there is no rock below the chassis but only below the solar array the clearance to rocks can be larger than the clearance to regolith.

Other metrics of interest include for instance the number of iterations which were needed, and the observed tranquilization time at the end of each alignment. Also, post processing of the obtained final orientations of the normal to the solar array allows the creation of maps showing for example the average power expectancy over a Phobos day at the end of the sequence.

Results—See Figure 15 for an overview of these results. Regarding the first two metrics, a general trend can be observed, that the Sun pointing algorithm generally trades stability for improved alignment. Further, perfect alignment is almost never achieved. This results from the Sun inclination at the planned landing sites and dates being usually lower than the safe tilt capability of the rover. Regarding the ground clearance, the value measured results from the sinkage of the wheels, the local topography as well as the rover's orientation. When collisions do occur, they are much more likely to happen with a rock than with the ground. As the analyzed case corresponds to the autonomous phase of the rover, this is to be expected as no ground loop was possible to move the rover away from any obstacles. As the rover's shutters are still closed at this point, minor collisions are acceptable. The results of the post processing show the expected power available to the system as a function of the landing site and date with consideration of both the effects introduced by the robotic system as well as those resulting from the orbits of Mars and Phobos.

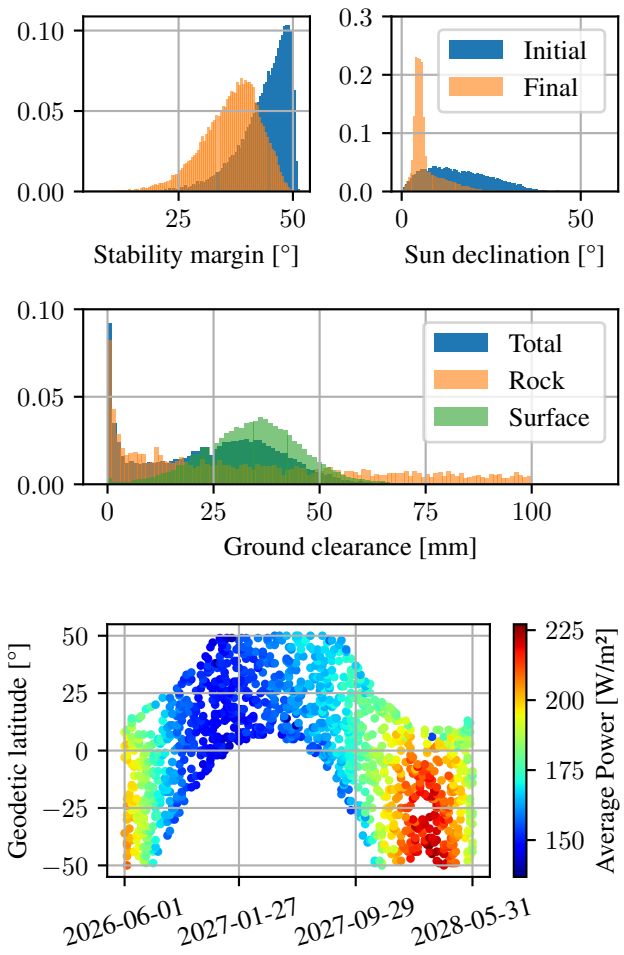


Figure 15. SKA simulation study results. Top left plot: comparison of rover stability margin before and after alignment. Top right plot: comparison of the alignment of the rover's solar panels with the Sun before and after alignment. Center plot: ground clearance of the rover after alignment, shown as total as well as separated as distance to rocks and surface. Bottom plot: available energy considering achieved pointing accuracy at different landing sites and dates.

4. Conclusion

This paper presented the multi-physics simulations that are being used to develop some of the key functions of the MMX Rover. The setup as well as individual components of the rover simulation have been presented including the rover model itself, the environment generator, the models for interaction between rover and environment and the scripts to integrate higher order logic. Because of its comprehensiveness and versatility it is established among the tools used by the development team of CNES and DLR.

The rover driving on Phobos itself is viewed as of great scientific value, the corresponding "locomotion science experiment" is described in [4]. Here the data flow is going the other way around, the actual motion of the rover on Phobos along with engineering data will be used to validate our rover and environment interaction models for low gravity. We hope that the whole field of rover simulation and terramechanics will profit from it.

The whole rover simulation and terramechanics activities will benefit from that. Besides the first driving in milli-g, we will thereby achieve the first rover and terramechanics model that is validated in milli-g.

Up until then, the development and refinement of the simulation models will continue. Depending on the needs and challenges we will extend the simulator to other applications and interface it to other subsystems that need it for development, testing or validation. For example, it is currently planned to do a simulation campaign very similar to the one performed for the SKA system to analyze the sequence used to take measurements with the rover's Raman spectrometer RAX. Further into the future, we plan to use the simulator during the operations phase: to test command sequences prior to commanding them and for environment reconstruction.

References

[1] S. Ulamec, P. Michel, M. Grott, U. Böttger, H.-W. Hübers, N. Murdoch, P. Vernazza, K. Özgür, J. Knollenberg, K. Willner, M. Buder, T. Hagelschuer, M. Grebenstein, S. Mary, J. Biele, C. Krause, T.-M. Ho, C. Lange, J. T. Grundmann, M. Maibaum, J. Reill, M. Chalon, S. Barthelmes, R. Lichtenheldt, F. Buse, R. Krenn, M. Smisek, J. Bertrand, C. Delmas, S. Tardivel, D. Arrat, N. Doumas, F. IJpelaan, L. Lorda, E. Remetean, M. Lange, O. Mierheim, S. Reershemius, T. Usui, M. Matsuoka, T. Nakamura, K. Wada, H. Miyamoto, K. Kuramoto, J. LeMaitre, L. Celine, A. Rafflegeau, C. Virmontois, H. Boirard, Y. Cho, and F. Rull, "A Rover for the JAXA MMX Mission to Phobos," in *Proceedings of the 70th International Astronautical Congress (IAC), 21–25 October 2019, Washington, DC, USA*, 2019.

[2] S. Tardivel and C. Lange, "The MMX rover: An innovative design enabling Phobos in-situ exploration," in *Proceedings of the Low-Cost Planetary Missions Conference (LCPM), 3-5 June 2019, Toulouse, France*, 2019.

[3] J. Bertrand, S. Tardivel, F. IJpelaan, E. Remetean, A. Torres, S. Mary, M. Chalon, F. Buse, T. Obermeier, M. Smisek, A. Wedler, J. Reill, and M. Grebenstein, "Roving on Phobos: Challenges of the MMX Rover for space robotics," in *Proceedings of 15th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA), 27-28 May 2019, Noordwijk, The Netherlands*, 2019.

[4] F. Buse, S. Barthelmes, M. Chalon, V. Langofer, W. Bertleff, R. Lichtenheldt, J. Skibbe, M. Bihler, R. Holderried, J. Reill, B. Vodermayer, L. Stubbig, R. Bayer, P. Vernazza, N. Murdoch, S. Ulamec, and P. Michel, "Wheeled locomotion in milli-gravity: A technology experiment for the MMX Rover." in *Proceedings of the 72th International Astronautical Congress (IAC), 25-29 October 2021, Dubai, United Arab Emirates*, 2021.

[5] S. Barthelmes, M. Bihle, B. Chalon, H.-J. Sedlmayr, R. Bayer, K. Sasaki, J. Skibbe, V. Langofer, R. Lichtenheldt, L. Stubbig, R. Holderried, S. Moser, F. Hacker, B. Vodermayer, W. Bertleff, T. Bahls, and F. Buse, "MMX rover locomotion subsystem – development and

testing towards the flight model," in *Proceedings of the 2022 IEEE Aerospace Conference, 5-12 March 2022, Big Sky, MT, USA*. Piscataway, NJ, USA: IEEE, 2022, accepted for publication.

[6] B. H. Wilcox and R. M. Jones, "The MUSES-CN nanorover mission and related technology," in *Proceedings of the 2000 IEEE Aerospace Conference, 18-25 March 2000, Big Sky, MT, USA*. IEEE, 2000, pp. 287–295.

[7] F. Zhou, R. E. Arvidson, K. Bennett, B. Trease, R. Lindemann, P. Bellutta, K. Iagnemma, and C. Senatore, "Simulations of Mars Rover Traverses," *Journal of Field Robotics*, vol. 31, no. 1, pp. 141–160, 2014.

[8] W. Li, L. Ding, H. Gao, Z. Deng, and N. Li, "ROS-TDyn: Rover simulation based on terramechanics and dynamics," *Journal of Terramechanics*, vol. 50, no. 3, pp. 199–210, 2013.

[9] S. Van wal, R. G. Reid, and D. J. Scheeres, "Simulation of Nonspherical Asteroid Landers: Contact Modeling and Shape Effects on Bouncing," *Journal of Spacecraft and Rockets*, vol. 57, no. 1, pp. 109–130, 2020.

[10] M. Hellerer, S. Barthelmes, and F. Buse, "The DLR Rover Simulation Toolkit," in *Proceedings of the 14th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA), 20-22 June 2017, Leiden, The Netherlands*, 2017.

[11] Q. M. Lam, N. Stamatakos, C. Woodruff, and S. Ashton, "Gyro Modeling and Estimation of Its Random Noise Sources," in *Proceedings of the 2003 AIAA Guidance, Navigation, and Control Conference and Exhibit, 11-14 August 2003, Austin, TX, USA*, 2003.

[12] Z. Miao, F. Shen, D. Xu, K. He, and C. Tian, "Online estimation of Allan variance coefficients based on a neural-extended Kalman filter," *Sensors*, vol. 15, no. 2, pp. 2496–2524, 2015.

[13] F. Buse, "Using superposition of local soil flow fields to improve soil deformation in the DLR Soil Contact Model - SCM," in *Proceedings of the 5th Joint International Conference on Multibody System Dynamics (IMSD), 24-28 June 2018, Lisbon, Portugal*, 2018.

[14] R. A. Jacobson and V. Lainey, "Martian satellite orbits and ephemerides," *Planetary and Space Science*, vol. 102, pp. 35–44, 2014.

[15] I. Parberry, "Designer worlds: Procedural generation of infinite terrain from real-world elevation data," *Journal of Computer Graphics Techniques (JCGT)*, vol. 3, no. 1, pp. 74–85, 2014.

[16] T. Archer, "Procedurally generating terrain," in *Proceedings of the 44th Annual Midwest Instruction and Computing Symposium (MICS), 8-9 April 2011, Duluth, MN, USA*. University of Wisconsin, 2011, pp. 378–392.

[17] M. K. Shepard, B. A. Campbell, M. H. Bulmer, T. G. Farr, L. R. Gaddis, and J. J. Plaut, "The roughness of natural terrain: A planetary and remote sensing perspective," *Journal of Geophysical Research: Planets*, vol. 106, no. E12, pp. 32 777–32 795, 2001.

[18] S. Tardivel, J. Biele, F. Buse, A. Hoerdt, R. Lichtenheldt, P. Michel, O. Kazunori, S. Schroeder, K. Willner, F. Wolff, and R. Ziese, "Phobos Environment Requirement Document for the MMX Robver Mission," Deutsches Zentrum für Luft- und Raumfahrt and Centre National d'Études Spatiales, Tech. Rep., 2020.

[19] J. Skibbe, S. Barthelmes, and F. Buse, "Locomotion Control Functions for the Active Chassis of the MMX Rover," in *Proceedings of the 2021 IEEE Aerospace Conference, 6-13 March 2021*. Piscataway, NJ, USA: IEEE, 2021.

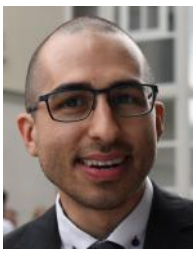
[20] F. Buse and T. Bellmann, "General Purpose Lua Interpreter for Modelica," in *Proceedings of the 14th International Modelica Conference, 20-24 September 2021*. Linköping University Electronic Press, 2021.

[21] S. Kümper, M. Hellerer, and T. Bellmann, "DLR Visualization 2 Library - Graphical Environments for Virtual Commissioning," in *Proceedings of the 14th International Modelica Conference, 20-24 September 2021*. Linköping University Electronic Press, 2021.

[22] M. Vayugundla, T. Bodenmüller, M. J. Schuster, M. G. Müller, L. Meyer, P. Kenny, F. Schuler, M. Bihler, W. Stürzl, B.-M. Steinmetz, J. Langwald, A. Lund, R. Giubilato, A. Wedler, R. Triebel, M. Smíšek, and M. Grebenstein, "The MMX Rover on Phobos: The Preliminary Design of the DLR Autonomous Navigation Experiment," in *Proceedings of the 2021 IEEE Aerospace Conference, 6-13 March 2021*. Piscataway, NJ, USA: IEEE, 2021.

BIOGRAPHY

Fabian Buse *received the degrees of B. Sc. and M. Sc. from RWTH Aachen University. Since 2015, he has been Research Associate at Institute of System Dynamics and Control (SR). His research interests are in terramechanics for planetary rovers. He is the lead engineer of the DLR Terramechanics Robotic Locomotion Lab (TROLL) and is leading the rover simulation for the MMX rover project.*



Antoine Pignède *received a double degree in electrical engineering and cybernetics from the Technical University of Darmstadt and the Norwegian University of Science and Technology in Trondheim in 2016. Since October 2016, he has been a member of scientific staff of the Institute of System Dynamics and Control (SR) at the German Aerospace Center DLR. His research area concerns modeling, simulation and optimization of planetary exploration rovers. In addition to the MMX rover, he is also doing simulations for the development of the DLR Scout rover with rimless wheels.*

Sébastien Goulet *is a Supaero engineer and holds a master degree in space sciences from UPS Toulouse III. Since 2006, he works at CNES as a CS Group contractor on mission planning design, simulation and studies for various satellite missions.*

Jean Bertrand *is graduated from Civil Aviation engineer school as electronic engineer. He had several positions at CNES since 1999, as qualification and radiation expert for VLSI components, responsible for several space equipment design, and since 2014, responsible of the mechatronic lab for space robotics. He is involved in Rover MMX project as robotic engineer*