J Intell Robot Syst (2019) 93:461–494
https://doi.org/10.1007/s10846-017-0680-9

Towards Autonomous Planetary Exploration

The Lightweight Rover Unit (LRU), its Success in the SpaceBotCamp Challenge, and Beyond

Martin J. Schuster[1] · Sebastian G. Brunner[1] · Kristin Bussmann[1] · Stefan Büttner[1] · Andreas Dömel[1] · Matthias Hellerer[2] · Hannah Lehner[1] · Peter Lehner[1] · Oliver Porges[1] · Josef Reill[1] · Sebastian Riedel[1] · Mallikarjuna Vayugundla[1] · Bernhard Vodermayer[1] · Tim Bodenmüller[1] · Christoph Brand[1] · Werner Friedl[1] · Iris Grixa[1] · Heiko Hirschmüller[3] · Michael Kaßecker[1] · Zoltán-Csaba Márton[1] · Christian Nissler[1] · Felix Ruess[3] · Michael Suppa[3] · Armin Wedler[1]

Received: 15 September 2016 / Accepted: 26 July 2017 / Published online: 7 November 2017

Abstract Planetary exploration poses many challenges for a robot system: From weight and size constraints to extraterrestrial environment conditions, which constrain the suitable sensors and actuators. As the distance to other planets introduces a significant communication delay, the efficient operation of a robot system requires a high level of autonomy. In this work, we present our Lightweight Rover Unit (LRU), a small and agile rover prototype that we designed for the challenges of planetary exploration. Its locomotion system with individually steered wheels allows for high maneuverability in rough terrain and stereo cameras as its main sensors ensure the applicability to space missions. We implemented software components for self-localization in GPS-denied environments, autonomous exploration and mapping as well as computer vision, planning and control modules for the autonomous localization, pickup and assembly of objects with its manipulator. Additional high-level mission control components facilitate both autonomous behavior and remote monitoring of the system state over a delayed communication link. We successfully demonstrated the autonomous capabilities of our LRU at the SpaceBotCamp challenge, a national robotics contest with focus on autonomous planetary exploration. A robot had to autonomously explore an unknown Moon-like rough terrain, locate and collect two objects and assemble them after transport to a third object – which the LRU did on its first try, in half of the time and fully autonomously. The next milestone for our ongoing LRU development is an upcoming planetary exploration analogue mission to perform scientific experiments at a Moon analogue site located on a volcano.

Keywords Autonomous mobile robots · Planetary exploration · Robotic challenge · Navigation · Manipulation · Autonomous task execution

Mathematics Subject Classification (2010) 68T40 · 70B15 · 93C85 · 68T45

1 Introduction

The *Lightweight Rover Unit (LRU)* [72] is particularly suited for planetary exploration. This field of application challenges the design of a robot in many aspects. Economic transportation to the planet forces the rover to be light. After arriving at the planet's surface, all sensors and actuators

This work was supported by the Helmholtz Association, project alliance ROBEX (contract number HA-304) and partially funded by the DLR Space Administration.

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s10846-017-0680-9) contains supplementary material, which is available to authorized users.

✉ Martin J. Schuster
martin.schuster@dlr.de

[1] Robotics and Mechatronics Center (RMC), Institute of Robotics and Mechatronics, German Aerospace Center (DLR), Münchener Str. 20, 82234 Weßling, Germany

[2] Robotics and Mechatronics Center (RMC), Institute of System Dynamics and Control, German Aerospace Center (DLR), Münchener Str. 20, 82234 Weßling, Germany

[3] Roboception GmbH, Kaflerstr. 2, 81241 München, Germany

need to work under the alien conditions. Even when faced with heavy communication delay and blackouts, the ground station team must be able to interact with the rover on a high level. As the delay renders teleoperation inefficient, the rover has to solve most tasks autonomously. It has to navigate unknown, rough terrain to explore the area and reach scientifically relevant locations safely. Therefrom, the rover has to manipulate the environment to take samples or to assemble technical equipment.

We designed the LRU to cope with the challenges of planetary exploration. Its unique construction is particularly lightweight (approx. 40 kg) and thus economic to transport into space. The LRU solely relies on sensor concepts (*Inertial Measurement Unit (IMU)*, stereo cameras) which work in alien conditions and are currently employed in space missions [28]. The LRU's locomotion system can drive over rough terrain and is highly maneuverable due to its four independent wheels, each of them having individual steering and driving motors. A force-controlled manipulator on the back of the rover picks up and assembles objects. The autonomy of the LRU stems from a variety of software components. We developed and integrated modules for on-board self-localization in GPS-denied environments, local and global mapping, fast obstacle avoidance, path planning, autonomous exploration, object detection and pose estimation, manipulation planning, platform and manipulator control, inter-process communication, autonomous high-level task execution as well as for a ground station mission control and the communication to it over a restricted and delayed connection.

We evaluated the LRU's capabilities in the official *SpaceBotCamp 2015* challenge, which posed typical challenges of a planetary exploration mission. The robot had to find two known objects in an unknown Moon-like rough terrain and assemble them at a base station, as shown in Fig. 1. The rover had to fulfill these tasks in a single run. Communication to the rover was heavily delayed as well as for most of the time unidirectional, restricting the ground station team to merely monitor the system. Therefore, the rover needs to solve all tasks autonomously, meaning that it has to cope with an unknown environment using its on-board sensor data only. In this work, we present the LRU and explain how it solved the challenge under all these constraints fully autonomously and successfully completed the mission in a single run in just thirty minutes, half of the time limit.

Since our success at the SpaceBotCamp in 2015, we continued the development of the LRU to extend its capabilities and move further in the direction of future space missions. Our next major milestone will be a lunar exploration analogue mission as part of the *Robotic Exploration of Extreme Environments (ROBEX)* project. The LRU will have to demonstrate its abilities in a Moon-analogue rough-terrain outdoor environment on Mt. Etna, Sicily, Italy in summer 2017. In order to gather scientific data on the crust model of the Moon – or on the composition of Mt. Etna in the analogue mission – the task for our rover is to deploy

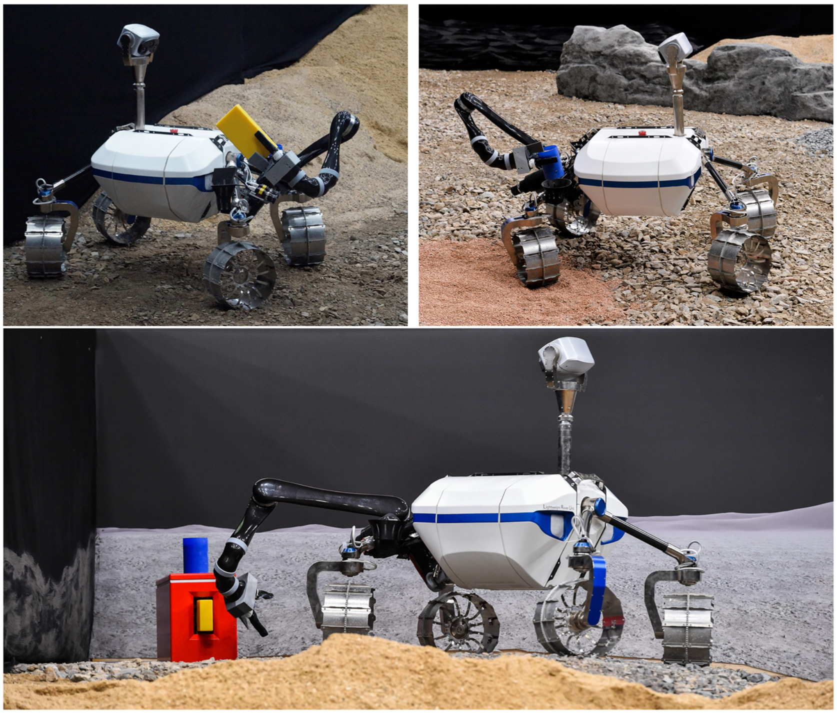


Fig. 1 Our Lightweight Rover Unit (LRU) picks up the battery (yellow, top left) and the sample (blue container, top right) and assembles them at the base station (red, bottom) during the SpaceBotCamp challenge

seismic measurement instruments. Similar to the SpaceBotCamp, these upcoming experiments will feature a limited communication channel to a far-away ground station and thus require a high level of autonomy of our robotic system.

This work constitutes a significantly extended version of our paper *"The LRU Rover for Autonomous Planetary Exploration and its Success in the SpaceBotCamp Challenge"* [58], for which we received the *IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)* 2016 best paper award. Its content goes beyond the original paper by both extending the sections of the conference paper as well as adding novel sections on our manipulator workspace analysis, the system's IT architecture, its power distribution unit, our next steps towards space qualification, our algorithms for autonomous exploration, manipulation planning, platform and manipulator control, our plan and world representations for autonomous mission control, our high-fidelity simulation environment and its usage as well as the upcoming ROBEX space analogue mission.

We start this work with a survey of related rover systems in Section 2 and then give an overview of the LRU in Section 3, detailing our robotic system's hardware components and highlighting related aspects. We present our software architecture in Section 4 and describe the LRU's individual software modules as well as our ground station mission control setup in its subsections. In Section 5, we introduce our high-fidelity Software-in-the-Loop (SiL) simulation that allows us to test a multitude of the LRU's software components early on in different simulated environments. Afterwards, we discuss our success in the SpaceBotCamp challenge in Section 6 and outline the upcoming ROBEX lunar exploration analogue mission in the subsequent Section 7. In our final Section 8, we summarize our contributions w. r. t. the LRU as a rover prototype for autonomous planetary exploration and provide an outlook on topics of ongoing and future work.

2 Related Work

The earliest space exploration rovers, *Lunokhod 1 & 2* (USSR), were landed on the Moon in 1970 and 1973. The relatively close proximity of the Moon allows for a teleoperation of lunar rovers from Earth. Also *Yutu* (China), landed in 2013, was mainly teleoperated from Earth, although it was capable of some autonomy for hazard avoidance and navigation. For missions on Mars, due to the long distance from Earth to Mars and the resulting high round-trip communication latency of eight to forty minutes, teleoperation becomes less feasible. Thus a high degree of autonomy of the rover system is necessary for efficient missions. The experiences gathered with the successful and ongoing Mars rover missions like the *Mars Exploration Rover Mission (MER)* [46] and the *Mars Science Laboratory (MSL)* [28] as well as earlier considerations on rover autonomy [69] clarify requirements and space-suitable options regarding hardware as well as software components. Autonomous navigation solutions for unstructured and unknown environments taking robot safety, resource management (e. g., power consumption) and general robustness into account are available in many robotic research areas such as autonomous driving, e. g., [67, 76], search and rescue [57] and planetary rovers / field robotic systems tested on Earth [26, 43, 60, 65, 73]. These systems, including all systems participating in the SpaceBotCamp 2015 challenge, use a variety of optical sensors for navigation, most commonly laser scanners and active RGB-D cameras [14, 66] or a combination of those, e. g., [32, 60, 65]. In contrast, in recent planetary rover missions like MER and MSL, the rovers only perceive with stereo vision systems for obstacle avoidance and navigation and use other types of cameras for a multitude of scientific purposes. For example, *Curiosity* [28], the latest and most advanced Mars rover, employs a total of 17 cameras: Two front and two back stereo hazard camera pairs (*HazCams*), two navigation stereo pairs (*NavCams*), one mast camera stereo pair to capture panoramic images of the Mars surface, a camera attached to a robot arm (*MAHLI*), a camera to control the descent system (*MARDI*) and an optical chemical measurement instrument (*ChemCam*). The advantages of camera systems are the availability of both, mature algorithms and compact, low-power flight-qualified cameras, whereas flight-qualified versions of other sensors for navigation and mapping, like suitable laser scanners, are currently not available [28, 46]. Sensor-specific noise characteristics and a typically less dense depth reconstruction of passive stereo camera systems pose a higher challenge on navigation and mapping algorithms compared to laser scanner-based systems [26, 60, 65], which allow high-precision measurements within a longer range of distances. Similar to past and current Mars exploration missions, the rover described in [73] as well as our own system only employ stereo camera systems. We use a single camera pair for navigation and mapping and another one for close-range object pose estimation for autonomous manipulation. For the SpaceBotCamp scenario, we added an additional color camera for long-range color-based object detection, but we can easily replace it by another type of scientific camera for mission-specific purposes. We refer to further related work concerning the individual aspects of our system in the respective sections throughout this work.

3 System Overview

Our system setup, presented in Fig. 2, consists of the LRU rover as well as a ground station to monitor the robot's

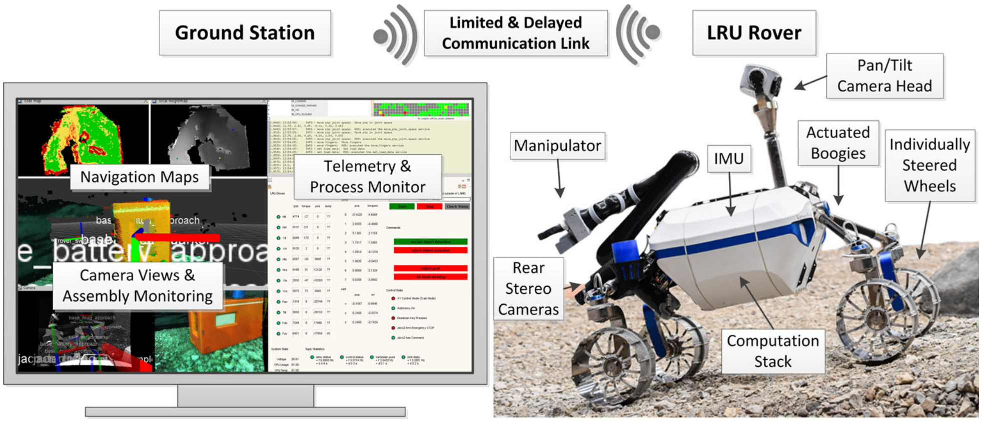

Ground Station
Limited & Delayed Communication Link
LRU Rover
Pan/Tilt Camera Head
Navigation Maps
Telemetry & Process Monitor
Camera Views & Assembly Monitoring
Manipulator
IMU
Actuated Boogies
Individually Steered Wheels
Rear Stereo Cameras
Computation Stack

Fig. 2 System overview showing the key components of the ground station GUI for monitoring and the rover hardware modules

state and to provide remote access for shared autonomy approaches. In the SpaceBotCamp scenario, described in more detail in Section 6, the communication link was artificially delayed by two seconds in each direction, which approximates the delay between a ground station on Earth and a rover on the Moon. In the following, we give details on the rover's hardware and present its software components in Section 4. We designed the LRU for operation in rough terrain by equipping it with four individually powered and steered wheels, which are connected to the rover body via two actuated bogies. At its front, the LRU rover has a pan/tilt camera head for navigation and object search. At its back, we mounted a manipulator for object pickup and assembly as well as an additional pair of stereo cameras for precise object pose estimation during manipulation.

3.1 Kinematics

The LRU has a total length of 1090 mm and a total width of 730 mm. It is designed to drive in rough terrain at a maximum velocity of 1.1 m/s. In Fig. 3, we give an overview of the LRU platform. It consists of four individually powered and steered wheels, attached to two bogies that each make use of a *Serial Elastic Actuator (SEA)*. That way, the rover features active and passive suspension in both bogies. The active element allows controlling the bogie's rotational position to adjust the center of mass in order to distribute wheel load in rough terrain or improve stability on steep slopes. The active-passive combination allows to implement an active dynamic body damping. Also all components attached to the rover body benefit from that additional degree of freedom by increasing their workspace, e. g., by repositioning the attached camera beam for a better view of the environment. Each actuator in the SEA, the wheel hub and steering drives utilizes an ILM38 drive train that was designed for space applications, see Section 3.2. Compared to other rover concepts, the LRU profits from its individually steered wheels, which allow for driving sideways and turning in place, thus increasing maneuverability. Furthermore, our four-wheeled kinematic has advantages in terms of compactness and weight compared to six-wheeled rovers.

3.2 Locomotion

As for the kinematics, reliability and robustness in rough terrain are also most important for the *Locomotion Sub-System (LSS)*. Due to their high peak torque and very high torque per volume and weight ratio, we use permanent magnet synchronous motors [70] in every rover joint. *Brushless DC (BLDC)* motors use fewer mechanical components and show no mechanical friction or cold welding risk because of brushes. All drawbacks of brushes like brushwear, electromagnetic interference due to brush-sparking and need for lubrication or atmosphere are avoided completely. Furthermore the heat dissipation is improved compared to conventional DC motors. The windings are on the stator, which has a direct conductive heat path to the exterior. Two actuator unit sizes were developed to meet the different requirements. We employ the small ILM25 unit to move the pan/tilt unit and the bigger ILM38 unit for wheel traction and steering. The name indicates that it is an internal rotor motor with the number showing the diameter of the motor stator in millimeters. Originally the ILM38 unit (rated torque: 5 Nm), shown in Fig. 4, has been developed for the *Mobile Payload Element (MPE)* rover prototype [29] and was developed further to the LRU actuator module. We employ it for traction as well as for steering and actuating the serial elastic module. To actuate the rover's pan/tilt unit [71], the concept was scaled to a smaller ILM25 module (rated torque: 2.4 Nm). This unit is actually used and validated in an ongoing DLR space mission called *Mobile Asteroid Surface Scout (MASCOT)* [50, 51]. Concerning the sealing of the drive units, we carried out a closer investigation. We tested the motor unit with different combinations of shaft surface coatings and *Polytetrafluoroethylene (PTFE)* seal elements. Several tests were executed in a thermal

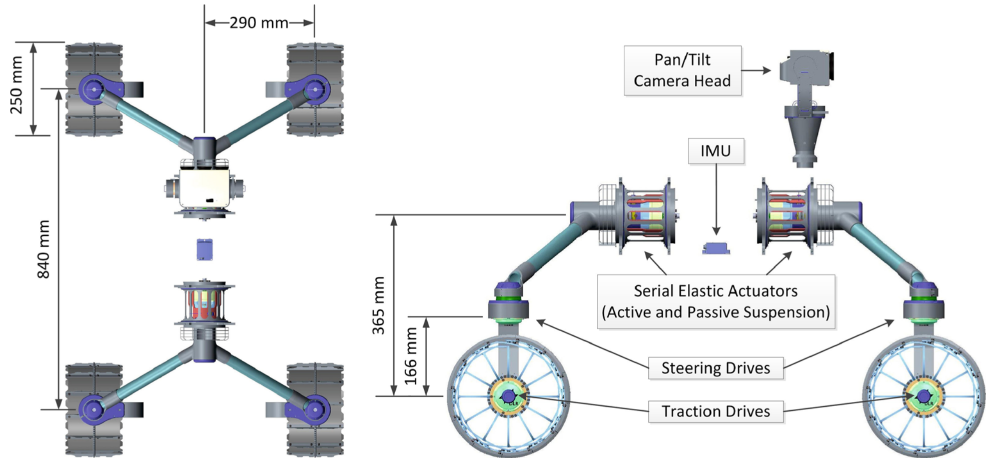


Fig. 3 LRU platform top and side view

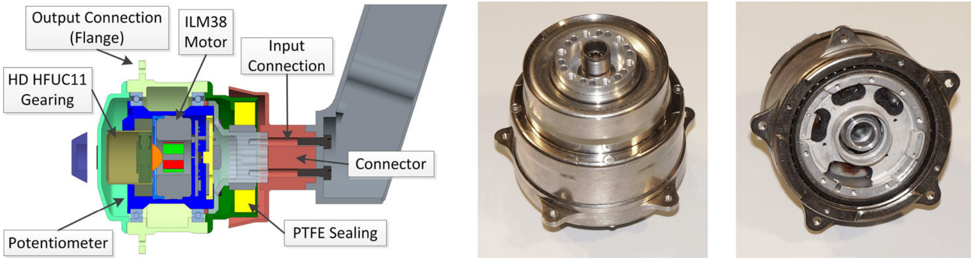
vaccum chamber to determine the best trade-off for low friction, good tightness and longterm stability. With a proper sealing solution, robust commutation electronics and space-proven lubrication, the drive unit can be considered as a component designed for space applications.

Our wheel design, as shown in Fig. 5, combines a flexible spring metal sheet running surface with a central rigid ring and thereby profits from both advantages of wide and narrow wheel types. The wheel-to-ground contact is focused on the rigid ring on hard and flat terrain, which leads to low rolling resistance. When driving in soft terrain, the flexible running surface ensures reduction of sinkage and, in combination with the grousers, gives the wheels maximum grip. We are able to adapt the stiffness of our elastic wheels by changing the number, geometry and material of the spokes, allowing them to absorb shocks directly in the wheels. In Fig. 5b and c, we present wheels with six blade springs (higher stiffness) and with three laser-sintered titan spokes (lower stiffness) respectively. The, more complex, laser-sintering production allows a precise definition of the elastic behavior in three dimensions based on finite-element analysis. While stiff behavior can increase the precision when grasping objects with the manipulator, lower stiffness helps to absorb shocks when driving in rough terrain.

3.3 Manipulator Selection and Adaptation

Fulfilling the goals of the SpaceBotCamp challenge required the robots to pick, place and manipulate objects. We considered three different manipulators to address these tasks: The in-house developed *Light Weight Robot III (LWR III)* [35], the *Jaco2* manipulator by *Kinova* [8] and the *P-Rob* manipulator by *F&P Personal Robotics* [11]. We performed an analysis of the workspace of the aforementioned manipulators using *Reachability maps* [24, 75]. For a comparison of effective dexterity and workspace volume, we employed high-resolution workspace models that we generated under consideration of all collisions between rover and manipulator as well as self-collisions, using the hybrid generation method introduced in [48].

In Table 1, we summarize the collected analysis data. We derived constraints on weight and payload capacity from the mechanical design and the SpaceBotCamp challenge description. Based on the assumption that general manipulation tasks are easier with a highly dexterous manipulator, we compare the workspace volume under different dexterity conditions: A $\geq$ 0 % dexterity condition reflects the full workspace volume that the manipulator can reach. The workspace gets smaller as we filter out volumes of lower

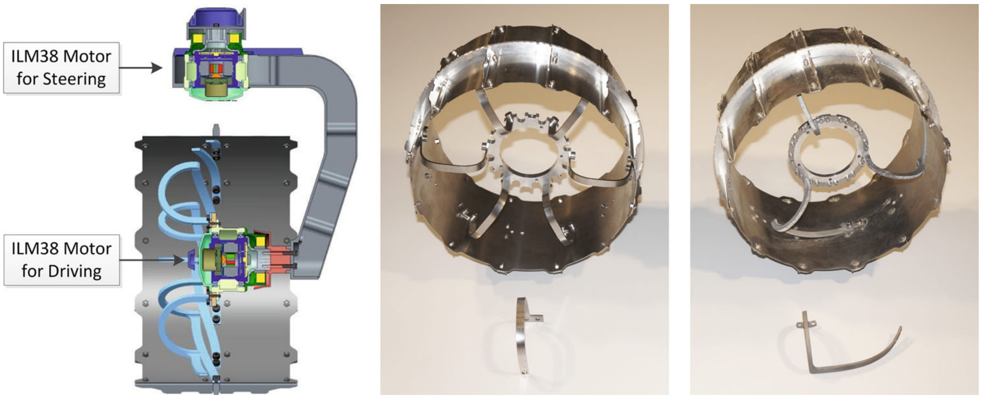


(a) Sectioned view of drive train with ILM38 motor unit

(b) Photo of ILM38 unit with closed casing

(c) Photo of ILM38 unit with open casing

Fig. 4 Drive train of the LRU's locomotion sub-system (LSS)



(a) Sectioned view of wheel hub

(b) Wheel with six blade springs (top) and single blade spring (bottom)

(c) Wheel with three laser-sintered titan spokes (top) and single titan spoke (bottom)

Fig. 5 LRU wheel hub and wheel spoke materials and configurations

than ≥ 50 % and ≥ 75 % dexterity. Applying this criterion, we observe that the *P-Rob* manipulator has a small usable workspace volume to begin with and significantly low dexterity in comparison to other manipulators. The *LWR* can operate in a large workspace volume with high dexterity, we however discarded it due to its heavy weight. We thus selected the *Jaco2* manipulator as it provides the most dexterous abilities throughout its workspace volume while satisfying our weight constrains.

Any potential collision of the manipulator with the rover itself decreases the effective workspace. Thus we searched for a mounting point that does not lead to significant reductions in workspace quality. Moreover, we aimed for maximizing the reach of the manipulator for objects on the ground. We thus generated multiple workspace models for different mounting pose of the manipulator, see Fig. 6 for an overview over variations of the mounting angle. While we model the manipulator with geometric primitives, i. e., cylinders and spheres, we represent the rover's body as a dense pointcloud of its hull, which we generated from its CAD model. Such a setup allows for computationally effective point-to-primitive collision checks during workspace generation, during which we employed a discretization with 1 cm voxel resolution and 6000 end-effector orientations in each voxel. We studied mounting poses in terms of the reach of the manipulator along a flat ground plane, its reach to the planned storage spaces on the rover's sides as well as its ability to reach the top of the rover's body for the possibility of additional object storage. We determined that a mounting angle of 45 deg is ideal w. r. t. the criteria listed above. Our analysis, depicted in Fig. 6, showed significant dexterity reductions due to collisions of the manipulator with the pan/tilt unit. We therefore decided to mount the manipulator at the back of the rover. To enhance the grasping capabilities of the six-degree-of-freedom manipulator, we optimized the gripper w. r. t. the shapes of the target objects. Further, we implemented a custom control framework for the *Jaco2* manipulator in order to actively control its contact forces during manipulation, see Section 4.5.2 for details.

3.4 Sensor Setup

In the context of space missions, it is essential to limit the weight and power consumption of the components to a minimum. Further, several test concerning heat, vibration, radiation, etc. have to be passed. Thus, pure camera-based systems are preferred to the integration of LIDAR systems, which typically are heavier, consume more power and are hard to qualify for space applications. Consequently, we based our complete autonomous navigation, mapping

Table 1 Manipulator weights, payloads and workspace volumes at three different dexterity thresholds ("≥ 0 %" indicates the full workspace)

Manipulator	Weight	Payload	Dexterity		
			≥ 0 %	≥ 50 %	≥ 75 %
LWR III	22 - 30 kg	7 - 14 kg	2.6673 m^3	1.8545 m^3	1.4518 m^3
P-Rob	20 kg	3 kg	1.5715 m^3	0.7525 m^3	0.2389 m^3
Jaco2	5.3 kg	1.5 - 2.5 kg	1.9505 m^3	1.8713 m^3	1.2746 m^3

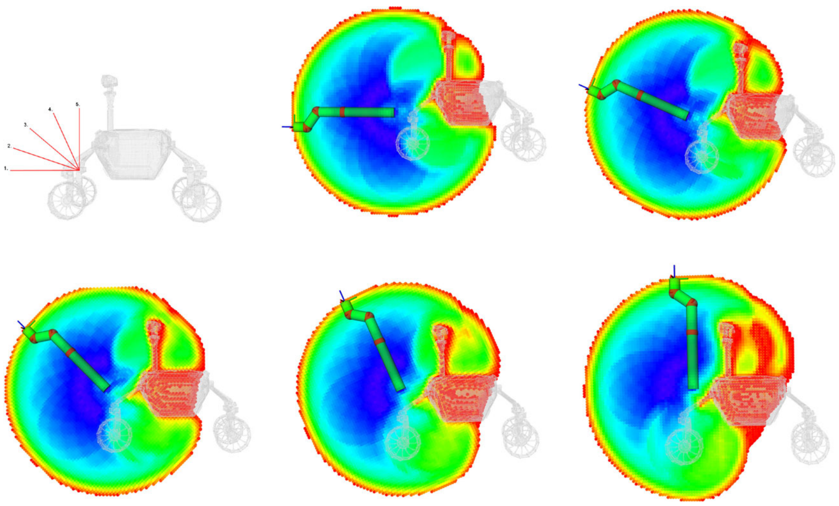


Fig. 6 Different mounting poses with the *Jaco2* manipulator's collision model in stretched configuration and a cross-section of the corresponding capability map

and exploration on a monochrome stereo camera system. Its baseline of 9 cm results from the demand for precise close-range data for navigation. We use an IMU in the rover's body to improve its ego motion estimation through sensor fusion. An additional center camera gives us color information for object detection. We however are able to exchange the center camera for other types of scientific cameras depending on the particular mission requirements. The cameras' fields of view are increased by using a pan/tilt mechanism that is able to pan the cameras ±180° and to tilt ±90° [71]. We employ an additional pair of color stereo cameras at the back of the rover to get unobstructed, high-resolution vision data for object pose estimation during manipulation. With a baseline of 6 cm, we adjusted these cameras to the manipulator's workspace and use them only during object pickup and assembly.

3.5 IT Architecture

In Fig. 7 we present an overview of our IT architecture, including the LRU rover, an on-site lander, our ground station as well as a channel simulator employed to simulate the delayed, constrained and lossy communication link between the ground station on Earth and both rover and lander residing far away on a foreign planet or moon. The role of the lander in our setup is limited to converting and forwarding the communication from the rover in a format appropriate for the transportation over the aforementioned link, as described in more detail in Section 4.7.2. The ground station contains a computer as the counterpart of the lander w. r. t. the communication link. It distributes the rover data to a set of operator workstations.

In order to build a truly autonomous system, we perform all required computation on board the LRU. As its main *On-Board Computer (OBC)*, we employ a standard industrial PC with an *Intel Core i7-3740QM* CPU (2.70 GHz) for all tasks that either process high-bandwidth data, like our vision-based perception, navigation and search & exploration pipeline, or that pose high computational demands like manipulation planning. In addition, we execute low-frequency tasks, such as high-level mission planning, on the LRU's main computer. We added a *Spartan-6* FPGA extension board to run the computationally intensive stereo matching. For the high-frequency, real-time control loops of the manipulator controllers, we employ an additional computer board with an *Intel Atom E3845* CPU (1.91 GHz) in order to separate them from the i/o-intensive image processing pipeline, thus helping us to satisfy the controller's real-time requirements. After the SpaceBotCamp, we also moved the, however less time-critical, platform control algorithms from the main computer to the Atom board, as indicated in Fig. 7. We connect the *Jaco2* manipulator to a separate *BeagleBone Black* computer board with an *ARM Cortex-A8* processor, which we extended through a custom-made cape in order to be able to address the manipulator through its high-frequency RS485 interface. While the computers are connected with each other via Ethernet, we employ *Ethernet for Control Automation Technology (EtherCAT)* for communication with the platform motor controller. It is a standardized, deterministic real-time bus that allows us to actuate the motors in sync with each other. The utilization of terrestrial components for computation hardware allows faster research cycles on novel software concepts. As stated above, we however run parts of our computationally intensive stereo image processing pipeline on an FPGA. The corresponding VHDL code thus could also be transfered and implemented on space-proof, i. e., radiation hardened, FPGAs in the future.

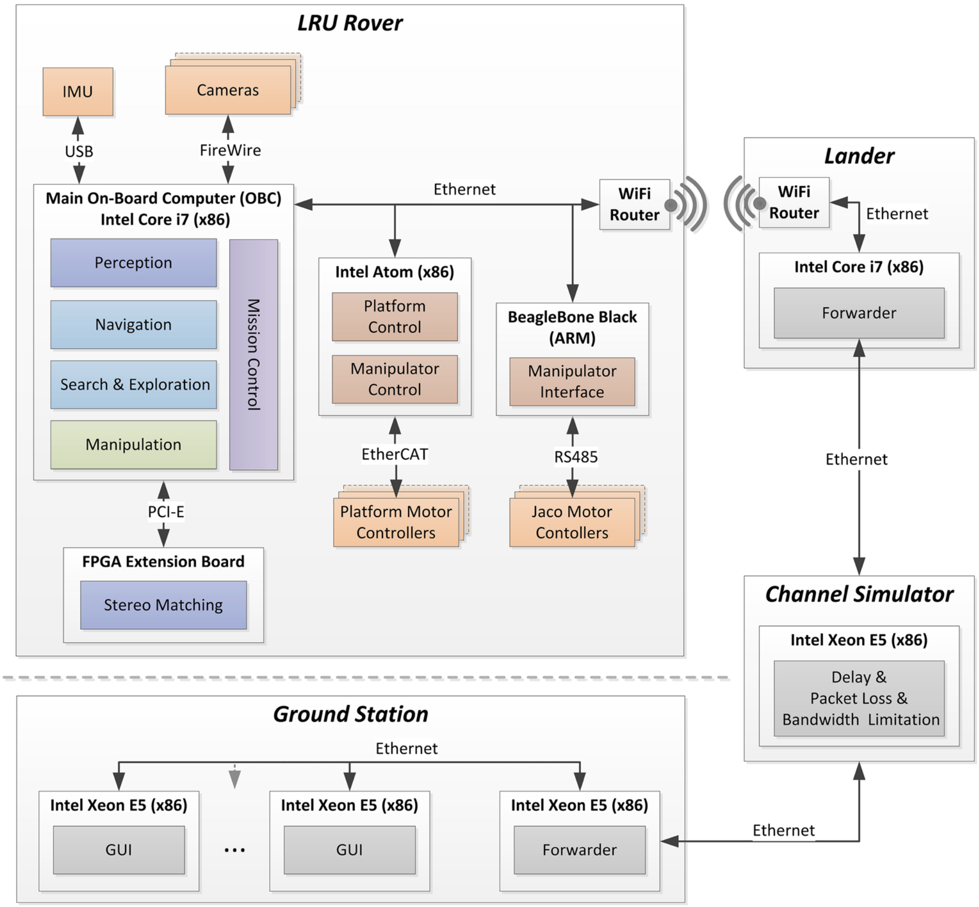


Fig. 7 IT architecture overview including the LRU's components as well as the communication link between an on-site lander and the ground station, which we artificially degrade in order to simulate a delayed and constrained link between Earth and foreign celestial bodies. All computation required for autonomous robot behavior is performed on-board the LRU

3.6 Power Distribution

An electro-mobility vehicle like the LRU demands for a robust and efficient energy management and distribution system, the so-called *Power Control and Distribution Unit (PCDU)*. Its main system requirements are reflected by the incorporation of a hierarchical safety design that is oriented towards the major safety guidelines for space technology, namely safety for the mission and safety for the equipment. According to these guidelines, the PCDU is responsible for the power supply of its payloads, uninterrupted choice of the best available power source and the delivery of telemetry data regarding the status of power sources and supply buses.

As depicted in Fig. 8, the PCDU receives the incoming power from either a pair of rechargeable Li-Ion batteries with a capacity of 208 Wh each or an external power supply. The supplied voltage can be in the range of 23 V to 30 V. A hierarchical power path controller selects the convenient power source according to its availability, i. e., the external power supply has the highest priority followed by the first and the second rechargeable battery. The PCDU incorporates a network of bus controllers that control and monitor the different power buses of the system. Each one of the bus controllers controls a bi-stable relay that feeds power either to the main bus or a connected sub-bus. In case of failures, e. g., over-current episodes on the sub-buses, the corresponding bus-controller can detect the failure and switch off the failing bus segment without interfering with the other power buses. The power is switched from the power path controller to the so-called main bus without any conversion by the main bus controller, which in turn serves as a central distribution rail for the following sub-buses:

- *Main OBC bus:* incorporates a DC/DC converter with 12 V output and powers the OBC and its subcomponents
- *High Voltage Payload (HVPL) bus:* includes a DC/DC converter for 24 V based payload, such as the *Jaco2* manipulator
- *Low Voltage Payload (LVPL) bus:* includes a DC/DC converter for 12 V based payload devices, such as external cameras, other sensors and computer boards

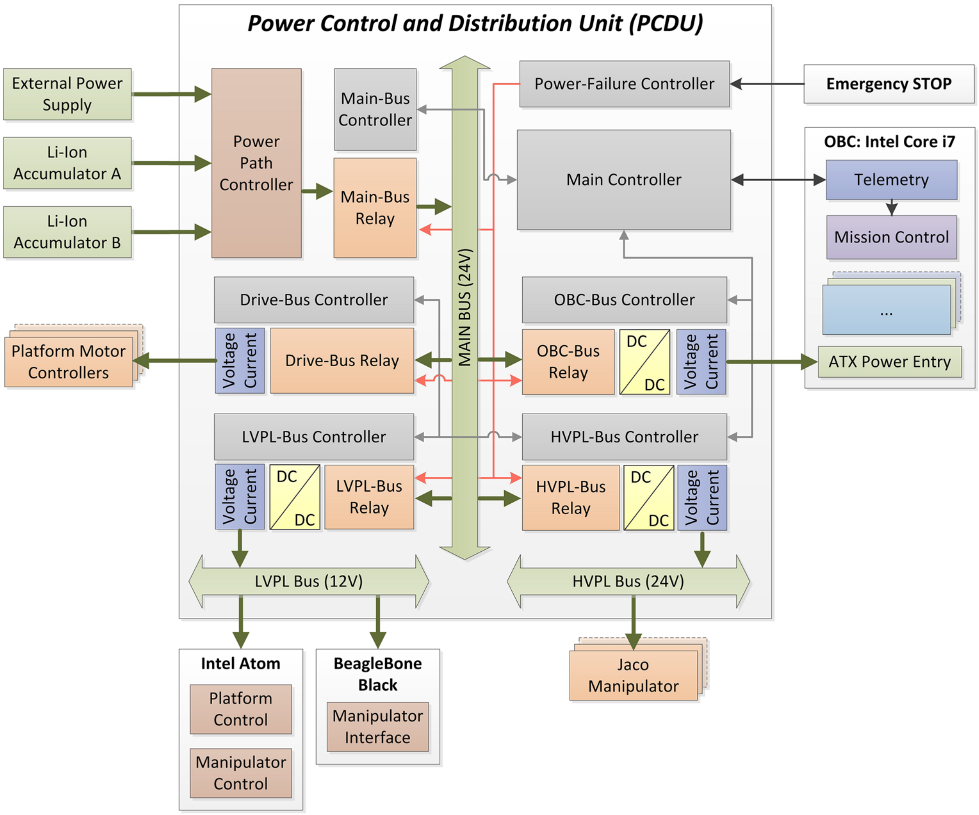


Fig. 8 The Power Control and Distribution Unit (PCDU) incorporates comprehensive switching of power buses and the automatic selection of the most appropriate power source. Mobile power sources (rechargeable Li-Ion batteries) can be hot-swapped. Telemetry data is fed to the on-board computer via USB

– *Drive bus:* receives and switches power directly from the main bus to supply the twelve drive controllers of the LRU

The PCDU main controller is responsible for collecting telemetry data from the bus controllers as well as for sending commands in order to achieve a safe and stable boot-up of the LRU system and to enable bus control capabilities for the OBC or a distant ground station. The transmitted telemetry data comprises information about all bus-related voltages and currents, status of batteries and status of all bus controllers. A further controller – the so-called power-failure controller – is independent from all other controllers and restores a stable boot condition in case of a complete power failure. For this purpose, the controller manages an independent power source that is charged as soon as power is stabilized on the LRU. In case of a power failure, all busses are switched off by this controller by redundant relay switching lines. Furthermore, this controller serves as a watch-dog for the emergency power-off button. The system's safety structure ensures that in the case of an electronics failure, the high level components, such as the OBC, are able to proceed their operation and that the autonomous task execution and failure handling modules as well as the ground station are notified and can decide on further steps. As the development of the PCDU along with an adaption of the LRU's infrastructure consumed more than a year, we did not yet test it during the SpaceBotCamp challenge, but integrated it for the ROBEX missions.

3.7 Next Steps to Space Qualification

The institutes of the *German Aerospace Center (DLR)* always focus on the aspect of space qualification of hardware and software developments when designing new robot prototypes in the context of space applications, which is also an important point during the development of our LRU. The LRU is a continuous development related to the lunar lander mission of the *European Space Agency (ESA)*. For this mission, the German space industry has developed a small sample-retrieval mobility unit, named *Mobile Payload Element (MPE)*, to collect soil samples and inspect the environment near a pol-landed lander [29]. During this project, the *Robotics and Mechatronics Center of DLR (DLR-RMC)* held the responsibility for both the *Locomotion Sub-System (LSS)* as well as for the *Autonomous Payload Element (APE)*, a perception and processing unit. We implemented these key modules as part of our LRU with the aim of qualifying them further to reach the space readiness level, which is relevant for future mission considerations. In the space sector, the *Technology Readiness Level (TRL)* describes the development status of single components as well as full systems. The MPE rover development, lead by the *OHB System AG*, has reached an overall level of TRL 3, i. e., the state of a proof of concept. The aim of the DLR-RMC qualification activities is to lift the modules of the LSS and APE to the stable state of TRL 6, which requires the modules to be tested in a relevant environment. This is the level from which on modules are considered for application in future space missions [50, 70–72].

Another interesting development parallel to the official ESA mission line is our support of the *Part-Time Scientists* team from Germany, Berlin, which takes part in the *Google Lunar X-Prize* competition [6]. In their *Asimov* and *Audi lunar quattro* rover prototypes [9], the same key elements regarding the LSS and APE are implemented as in the MPE and LRU prototypes. At DLR-RMC, we face the goal of implementing the latest developments in software and hardware to achieve a high performance and go beyond the current state of the art, while at the same time developing carefully chosen key components further, testing and validating them, in order to reach a level of robustness and reliability that makes them relevant to be employed in future official space mission experiments.

4 Software Architecture

We present an overview of our software architecture in Fig. 9 and describe its individual modules in detail in the following sections. We established the data flow between our components via three different middlewares in order to satisfy their particular needs: *Links and Nodes* for real-time control and *SensorNet* for the distribution of

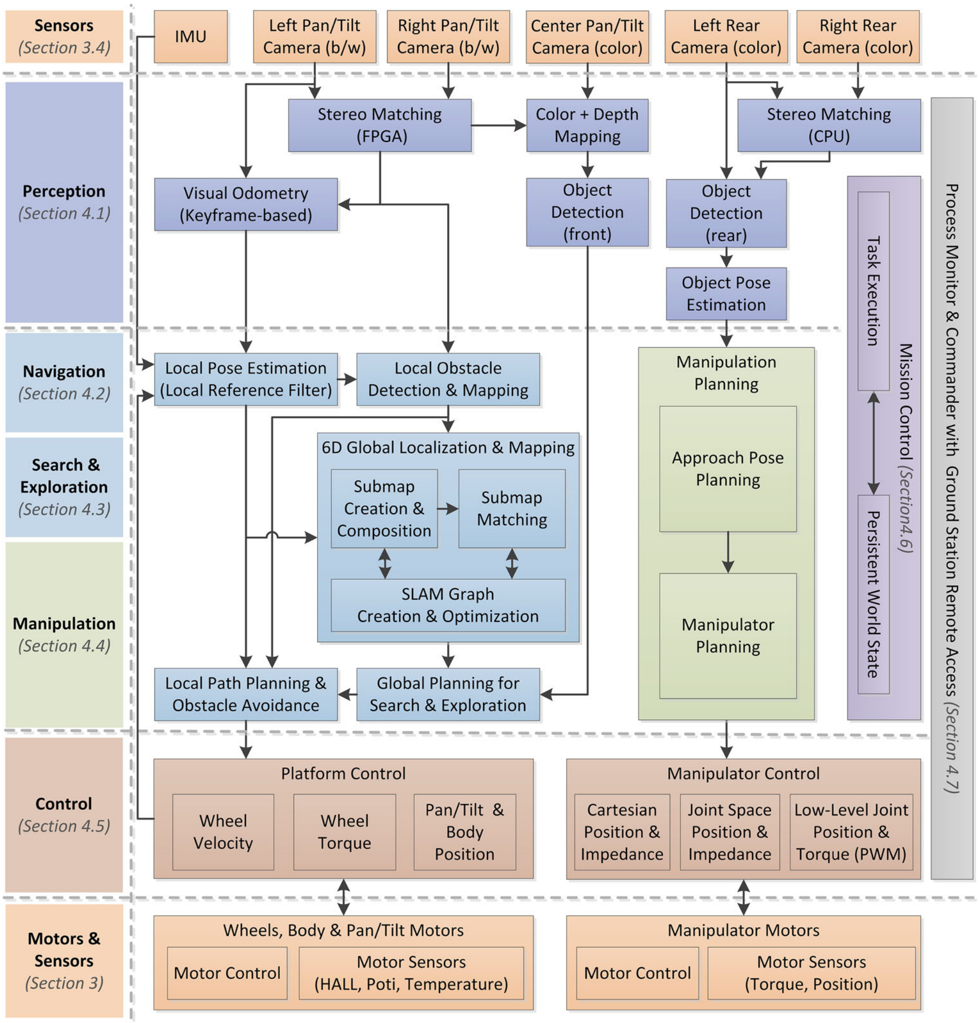

Sensors (Section 3.4)
IMU
Left Pan/Tilt Camera (b/w)
Right Pan/Tilt Camera (b/w)
Center Pan/Tilt Camera (color)
Left Rear Camera (color)
Right Rear Camera (color)
Perception (Section 4.1)
Stereo Matching (FPGA)
Color + Depth Mapping
Stereo Matching (CPU)
Visual Odometry (Keyframe-based)
Object Detection (front)
Object Detection (rear)
Object Pose Estimation
Navigation (Section 4.2)
Local Pose Estimation (Local Reference Filter)
Local Obstacle Detection & Mapping
Search & Exploration (Section 4.3)
6D Global Localization & Mapping
Submap Creation & Composition
Submap Matching
SLAM Graph Creation & Optimization
Manipulation (Section 4.4)
Manipulation Planning
Approach Pose Planning
Manipulator Planning
Local Path Planning & Obstacle Avoidance
Global Planning for Search & Exploration
Task Execution
Persistent World State
Mission Control (Section4.6)
Process Monitor & Commander with Ground Station Remote Access (Section 4.7)
Control (Section 4.5)
Platform Control
Wheel Velocity
Wheel Torque
Pan/Tilt & Body Position
Manipulator Control
Cartesian Position & Impedance
Joint Space Position & Impedance
Low-Level Joint Position & Torque (PWM)
Motors & Sensors (Section 3)
Wheels, Body & Pan/Tilt Motors
Motor Control
Motor Sensors (HALL, Poti, Temperature)
Manipulator Motors
Motor Control
Motor Sensors (Torque, Position)

Fig. 9 LRU software architecture: On-board key components and data flow

high-bandwidth vision data, both being developed at our institute. In addition, we connect our higher-level software components via the widely used *Robot Operating System (ROS)* [49].

On board the LRU, we typically run more than 100 software processes in parallel, involving the execution of about 100 different libraries and components developed by more than 20 internal developers. In order to manage this complexity, we employ the process manager of *Links and Nodes* to monitor process output, manage runtime dependencies and allow the compilation of mission settings by combining pre-defined modules and configurations. In addition, we developed our own release and dependency management toolchain *RM Package Management (RMPM)* to track and deploy consistent software versions to the LRU as well as to mockup systems and simulations. Furthermore, in order to obtain reproducible builds and raise the quality of our software components, we use a continuous integration workflow, supported by automated builds and tests on all code changes committed to our version control system. We thereby try to bridge the gap between the requirement for quick prototyping during research and the need for well-defined software development processes and standards, both to handle such a complex robot system in its current state as well as with regard to the direction of future space qualification.

4.1 Perception

The LRU's perception of the environment is purely vision-based. In this section, we describe our image processing pipelines, starting with stereo matching to generate 3D information, visual odometry as the key input for our navigation components and long-range object detection as well as close-range object pose estimation required for autonomous manipulation tasks.

4.1.1 Stereo Matching and Visual Odometry

We address our five cameras via our own middleware *SensorNet* that is based on shared-memory transport in order to stream high-bandwidth image data at low latency. For the pair of stereo cameras in the pan/tilt head, we perform dense stereo reconstruction through *Semi-Global Matching (SGM)* [33] running on an on-board Spartan-6 LX75 FPGA with a resolution of 1024 × 508 px at 14.6 Hz. We use the resulting depth data for stereo visual odometry, obstacle avoidance and 3D environment mapping. We chose monochrome cameras for our navigation stereo setup as they have a shorter exposure time as well as a higher effective resolution than color cameras. This is important for the accuracy of our visual odometry estimation [34], which constitutes the basis of the LRU's self-localization. Furthermore, we map the color information of our third pan/tilt camera onto the navigation cameras' depth data to serve as input for our object detection. The additional rear-facing color stereo cameras are only triggered on-demand for close-range precise 6D object pose estimation. We thus can perform their stereo reconstruction on the CPU at a higher resolution than supported by our FPGA. In order to achieve an accurate mapping of color to depth data as well as to obtain a reliable object localization for grasping, the transformations between the stereo and color cameras as well as between the rear cameras and the gripper have to be determined. We automated this process, called *hand-eye calibration* [64], by attaching a calibration pattern to the LRU's gripper to move it to pre-defined poses within the cameras' fields of view.

4.1.2 Object Detection

Detecting both near and far objects with a fixed-lens stereo setup is challenging as far-away objects likely appear too small in the image to identify shape properties. All the SpaceBotCamp objects however have large unicolored homogeneous surfaces on all sides. For far-range detection, we thus employ a color-based object segmentation, coupled with an estimation of the objects' extents, as outlined in Fig. 10. A learning-based classification approach allows us to achieve a robust segmentation in various light conditions: The feature vector $f(i, j)$ for each pixel at coordinates i, j consists of the pixel's hue, saturation and value (HSV) parameters for its own color and brightness

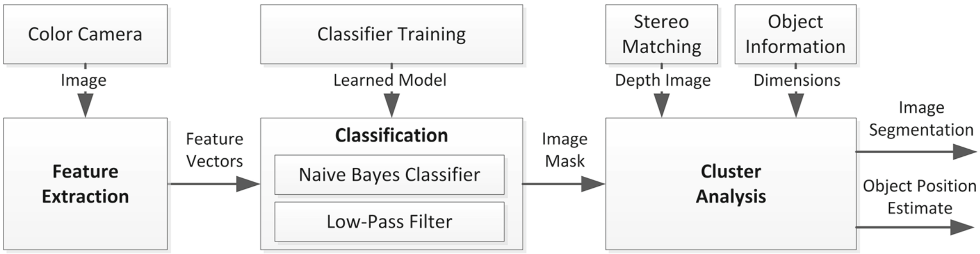


Fig. 10 Object detection pipeline: First we classify the pixels in the color image and then cluster them to object hypotheses. Depth information allows us to check these against the known object dimensions

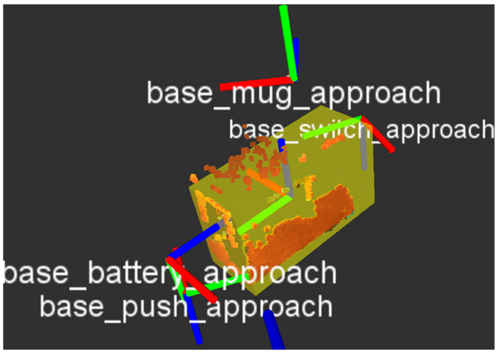


Fig. 11 Successfully estimated position and orientation of the base station object from a partial view and different task-relevant manipulator approach poses shown next to it (red: input point cloud, yellow: bounding box of object model)

as well as additional features describing its local neighborhood. Since a complete description of a pixel's neighborhood would exceed a reasonable complexity for its feature vector, we approximate the neighborhood relation by solely adding two further dimensions, which describe the mean intensity at two given neighborhood distances $n1$ and $n2$. Compared to a feature vector including a complete description of the neighborhood, we are thereby able to reduce the dimension from 78 to 5: $f(i, j) = \{h(i, j), s(i, j), v(i, j), v_{n1}(i, j), v_{n2}(i, j)\}$

During a training stage, we took a few thousand annotated images of the SpaceBotCamp objects and systematically varied light conditions, background and object poses. We then fed the resulting feature vectors into a Naïve Bayes classifier. After removing outliers and low-pass-filtering the mask, we combine the resulting segments with depth data for a cluster analysis. Since depth and angle w. r. t. the camera are sufficient to estimate an object's position and size, we filter all clusters according to the object's expected dimensions in order to obtain a reliable and robust estimation of its location. We experimentally determined a reasonable maximum distance of 5 m for object detection with the LRU's camera system, taking into account robustness and the cost of acting on false positives during the SpaceBotCamp challenge.

4.1.3 Object Pose Estimation

For object pickup and assembly tasks, the LRU not only needs to locate the objects, but also needs to estimate their 6D poses relative to the rover's manipulator with a certain precision. For example, the pose of the base station has to be estimated with an accuracy of up to 1 cm for a robust insertion of the battery object. We first capture the target object with the LRU's rear stereo cameras. However, the uniformly colored objects in the SpaceBotCamp scenario are challenging for stereo reconstruction, leading to poor and sparse depth images with a high level of noise. We therefore employ an additional structured-light projector to enhance the texture of these objects by projecting an artificial random pattern onto them. The pattern itself does not need to be calibrated as it only enhances the structure in the images and is not necessary for other, textured, objects. Nonetheless, the stereo data is still noisy and incomplete, as shown in the red point cloud in Fig. 11, which we successfully employed to estimate the pose of the base station. In Fig. 12, we outline the workflow of our object recognition pipeline.

As a pre-processing step, we use the information from the object detection to mask out the background in the depth image. Further, we downsample the point cloud computed from stereo matching, then smooth it and estimate surface normals using the *Point Cloud Library (PCL)* [52]. For the pose estimation itself, we employ our own extension of PCL's sample consensus module, which is capable of fitting more shapes (including boxes, as needed here). Compared to traditional *Random Sample Consensus (RANSAC)* methods, we thereby do not only optimize the inlier count probabilistically, but also minimize the percentage of the shape's volume that is contradicting the depth measurements [21].

The SpaceBotCamp base station object, however, is not always fully visible, thus the best-matching model is likely to be too small. Moreover, the base station is not a perfectly symmetric box, but has other features, such as the slot into which the battery had to be inserted. This poses two challenges for the fitting, which is designed only to handle primitives, like for example the battery object. We thus added a post-processing step to handle both the basic

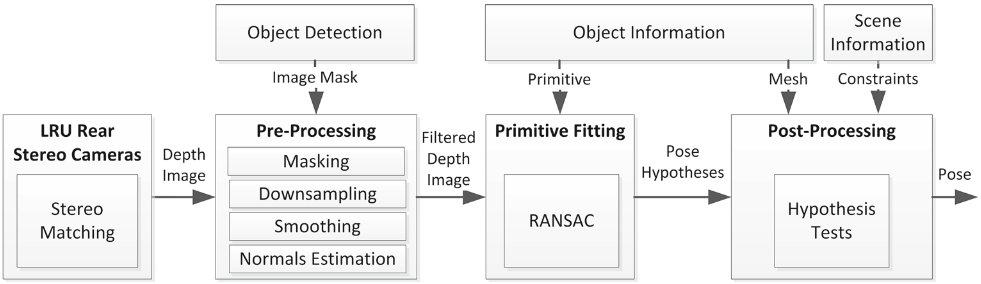


Fig. 12 Object pose estimation pipeline: First we pre-process depth images from the rear cameras using information from our object detection. We then apply primitive-fitting and finally find a feasible solution by comparing against a high-resolution 3D model and checking scene constraints

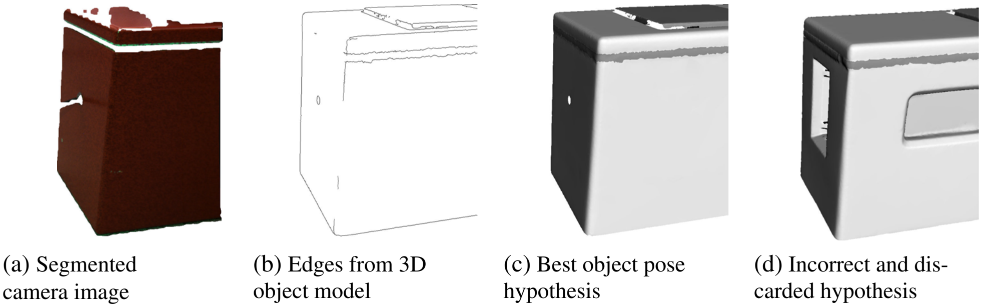

(a) Segmented camera image (b) Edges from 3D object model (c) Best object pose hypothesis (d) Incorrect and discarded hypothesis

Fig. 13 Different stages of the edge-based orientation disambiguation during object pose estimation of the SpaceBotCamp base station object

and more complex cases, consisting of an extension of our scene description language with the option to define a 3D model for the fitted shapes. We acquired a 3D mesh of the base station object through the automatic 3D modeling system described in [42]. This highly accurate model is then placed in all possible positions overlapping the fitted model to generate a set of 6D pose hypotheses. We thereby exclude those hypotheses that violate any constraints that we defined a priori in the scene description, for example stating that the base station cannot be standing upside down. We then render all model placement hypotheses and compare the resulting images with the camera image using an edge distance-based scoring, next to the volumetric information scoring from [21]. We give an example of the partially visible base station object and correct/incorrect box hypotheses in Fig. 13.

4.2 Navigation

In this section, we describe the components of our navigation stack, both for on-board local and global self-localization and mapping of an previously unknown semi- or unstructured environment. We designed our mapping framework to allow the LRU to operate in GPS-denied, previously unknown indoor as well as rough-terrain outdoor environments. We combine fast local mapping for obstacle avoidance with a submap-based online global mapping approach to create a consistent 3D environment model for search and exploration.

4.2.1 Fast Local Self-Localization and Obstacle Mapping

For a robust, real-time local pose estimation, we fuse our visual odometry estimates with wheel odometry and IMU measurements in a local reference filter [55], which is realized as a keyframe-based *Extended Kalman Filter (EKF)* with time-delay compensation [56]. As a first mapping step, we perform a fast stereo-error adaptive obstacle and terrain classification on the depth images from the pan/tilt stereo cameras, as we present in detail in [16]. The depth error of stereo reconstruction grows quadratically in distance to the cameras. As the association of 3D data and the original camera viewpoints is lost during its aggregation into submaps, we have to consider the stereo errors early on. We present a visualization of the 2.5D terrain classification maps created during our SpaceBotCamp run in Fig. 14. We can directly use these cost maps for local path planning and fast obstacle avoidance.

4.2.2 Online Global Self-Localization and 3D Environment Modeling

In addition, we integrate the full 3D stereo data, including the obstacle classification results, into submaps by aggregating it along the trajectories estimated by our local reference filter. We always switch the filter's frame of reference into the origin of the current submap in order to maintain consistency and numerical stability within the filter as well as to allow for a more accurate integration of

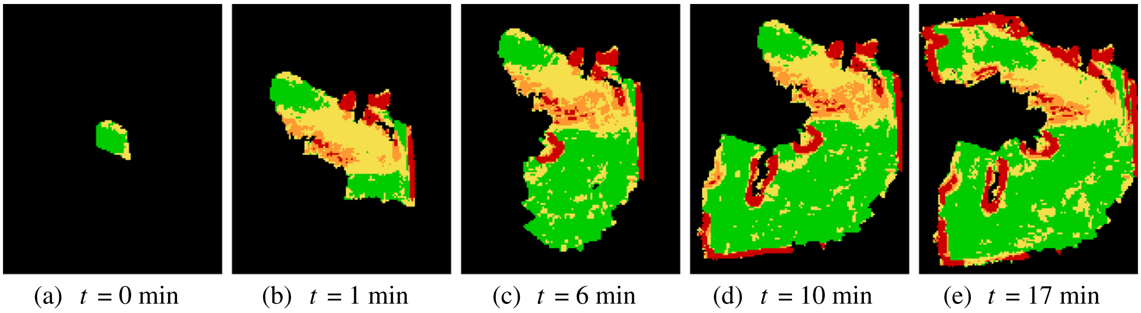

(a) $t = 0$ min (b) $t = 1$ min (c) $t = 6$ min (d) $t = 10$ min (e) $t = 17$ min

Fig. 14 Sequence of 2.5D terrain classification maps created during our SpaceBotCamp run

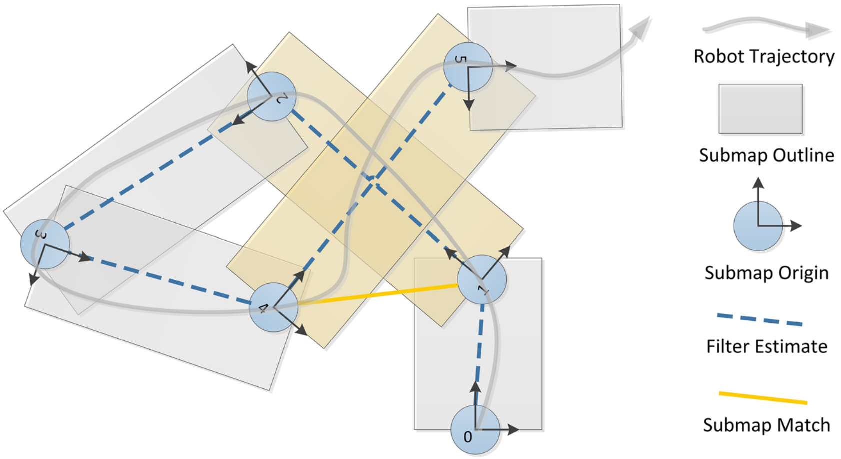


Fig. 15 Schematic of the SLAM graph for our submap-based global mapping. The highlighted rectangles represent overlapping submaps that match, resulting in a loop closure constraint that is added to the graph

the filter's uncertainty estimates into our overlying SLAM system. For online global optimization of pose and map estimates, we add the submap origins as nodes to a SLAM graph, connect them via the filter estimates and then run the iSAM2 [40] incremental least-squares error minimization after each change of the graph. The combination of a local reference filter and incremental graph SLAM allows us to benefit from their particular advantages: The filter provides real-time, long-term stable state estimation for control and fast obstacle avoidance while the online graph optimization provides global pose and map estimates. We provide a detailed description of our integration of local reference filters and global optimization as part of our modular SLAM system in [59].

In order to generate loop closure constraints, we perform 6D map matching using the obstacle classifications and the 3D submap data to compute relative transformations between pairs of submaps, as we describe in detail in [17]. Based on the most recent SLAM estimates for the pairwise transformations and uncertainties between submaps, our map matcher selects and ranks the most promising potentially matching pairs for registration in order to improve its efficiency over a brute-force approach. By relying on 3D geometric features instead of 2D image features, we achieve a higher robustness w. r. t. changing viewpoints and varying light conditions. After a final *Iterative Closest Point (ICP)* refinement step and outlier filtering, we integrate the resulting 6D transformations into our SLAM graph, as we sketched out in Fig. 15. We thereby weigh these relative transformations by the error estimates computed during map matching. We present a top-down and side view of a height-colored 3D point cloud map of the SpaceBotCamp scenario, as created by our mapping pipeline, in Fig. 16. Beside the 3D point data, we also visualized the underlying graph that is created and optimized by our SLAM system. As recent work beyond the SpaceBotCamp, in

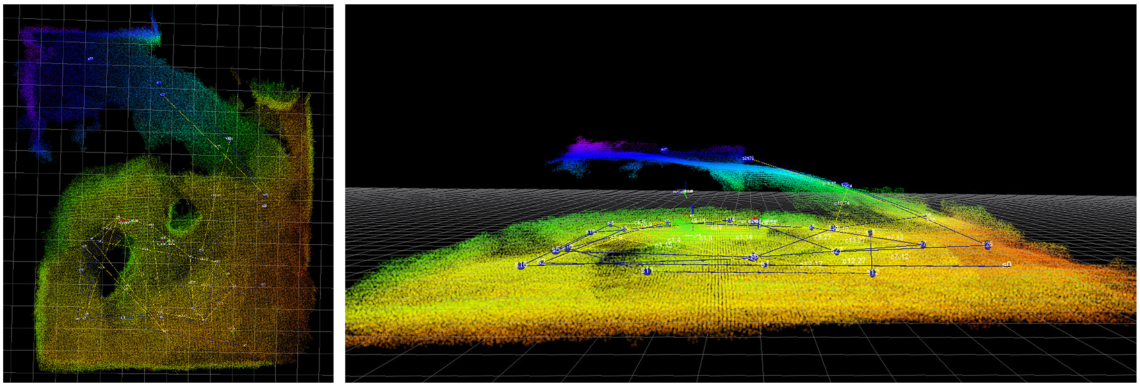


Fig. 16 Top-down and side view of height-colored 3D point cloud map of the SpaceBotCamp scenario created by our graph-slam components (blue ellipsoids: covariance of submap origin position estimates, blue and yellow lines: graph constraints from filter estimates and submap matches). The hill in the top left corner, see also Fig. 30, is clearly visible, its flank however is missing in the point cloud as it was too steep to be observed from above and the LRU did not look up to see it from below

addition to 3D point data, we attach 3D probabilistic voxel-based maps (resolution: 0.1 m) to each of our submaps and merge these when computing an estimate for the full map. We employ the freely available open-source OctoMap library [38] for the creation of these probabilistic voxel maps. They allow for an explicit representation of occupied, free and unknown space and are thus valuable input for our autonomous frontier-based exploration that we present in Section 4.3.2.

4.3 Search and Exploration

For planetary exploration, we consider previously unknown environments as well as scenarios where a low resolution map, e. g., from a satellite image, is available. As in real space missions, the rover should be able to execute predefined tasks fully autonomously, but also be able to support a semi-autonomous operation as low-level remote control is either impossible or very inefficient due to the aforementioned communication delays. Starting with a rough map from a low resolution image, a waypoint mission can be planned, either in order to fully explore the whole territory or to explicitly focus on areas of interest where relevant targets are likely to be found. In scenarios where no prior information is available, we carry out a frontier-based exploration.

4.3.1 Object Search

In the SpaceBotCamp challenge, a rough map (1 px ≙ 0.5 m) was given, however we did not know the location of the target objects a priori. Thus, in order to find all targets, we needed to plan waypoints such that they cover the complete field. We therefore divided the area into grid cells sized according to the LRU's range of reliable perception. During the search and exploration in the challenge, we scanned the area on each waypoint with our pan/tilt sensor head, covering a full 360° angle of view around the rover in order to detect objects. Unreachable waypoints are approached as close as possible and then skipped, i. e., the motion is stopped and the full 360° view is conducted. If an object is detected, the rover interrupts its current exploration state to pick up the object and continues with the next waypoint afterwards. We apply a fast, graph-based 2D path planner for the LRU to navigate autonomously between waypoints, taking the surrounding obstacles and local terrain classification maps into account.

4.3.2 Exploration

In unknown scenarios, we employ a frontier-based exploration algorithm [74] in order to maximize coverage. In comparison to the aforementioned object search with a-priori knowledge, for which we plan waypoints offline, our autonomous exploration plans local exploration goals online. At each frontier to unexplored territory, we find candidate exploration goals with respect to the current global pose and map estimates, as sketched out in Fig. 17. In order to decide, which goal shall be approached next, the utility of reaching a goal is estimated. We assess this utility by taking the distance to the exploration goal and an expected information gain into account. To prevent the system from moving unnecessarily towards a dead end, we implemented a service that repetitively reevaluates the utility of reaching the current exploration goal.

In detail, we derive the frontiers from our local 2D occupancy grid map similar to [74], except that we split long frontiers into several separate frontiers in order to guarantee that the robot can observe the whole frontier from a possible exploration goal.

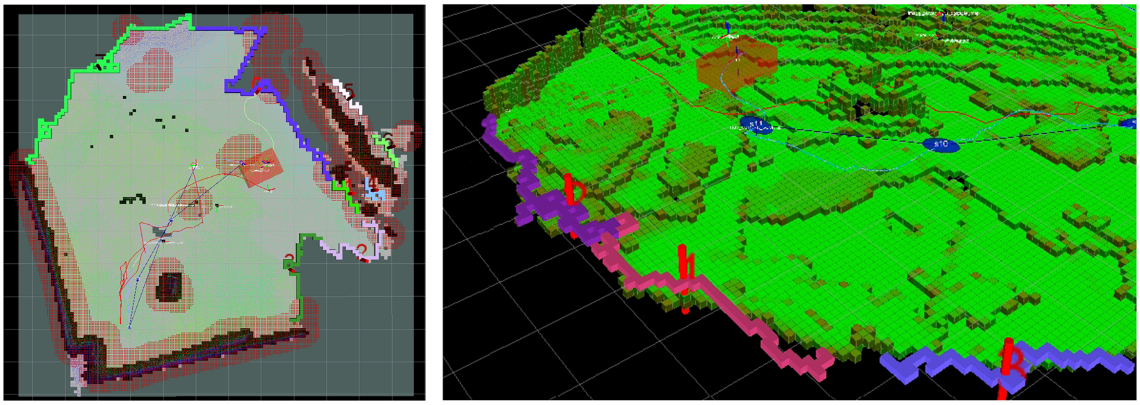


Fig. 17 Top-down and side view of the exploration process. We visualize the frontiers between free, known space and unknown space as clusters of colored frontier cells. At each frontier, a candidate exploration goal is represented by a red rod and labeled in sequence of its utility. The top-down view shows our 2D occupancy grid map, the local obstacle point cloud and the detected frontiers. The side view visualizes the 3D probabilistic voxel grid map used for information gain calculations and the detected frontiers

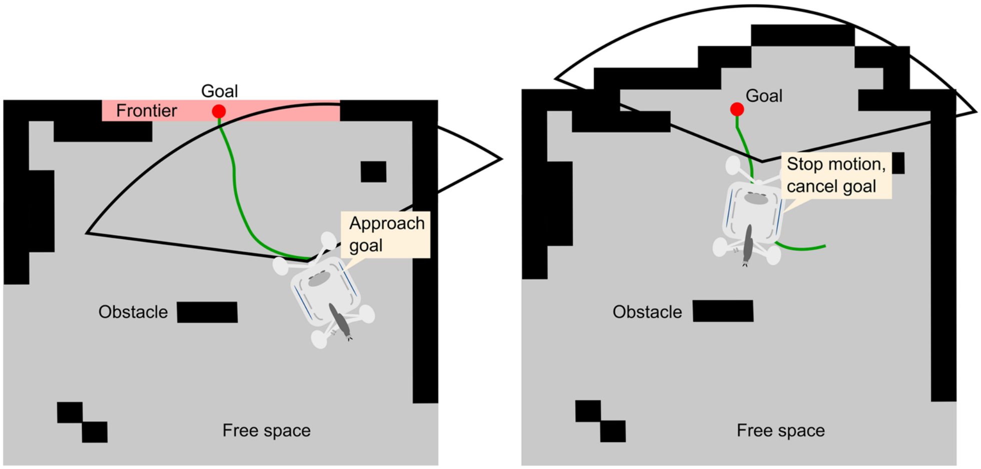


Fig. 18 Online task cancellation during exploration: The robot plans a path to the next goal and starts approaching it. During the approach, the utility of reaching the goal is repetitively rechecked. The robot detects that it will end up in a dead end that it has already fully explored and therefore stops the goal approach before reaching the exploration goal

At each frontier, we define the cell at the center of the frontier to be a candidate exploration goal g_i. At each g_i, we calculate the expected information gain IG_i for our stereo camera sensor readings from the 3D voxel maps created by our mapping system, as described in Section 4.2. Let $\mathcal{X}$ be the set of voxels that can be observed from g_i. The IG_i is defined as the sum of the differences between the current entropy H and the expected entropy $E[H']$ at each voxel $x \in \mathcal{X}$:

$$IG_i = \sum_{x \in \mathcal{X}} H(x) - E[H'(x)] \tag{1}$$

H is calculated using the probability of occupancy of the current voxel space. To estimate $E[H']$, we simulate the stereo camera at each g_i and perform a sparse ray tracing, taking into account a decreasing probability of observability with increasing distance from the stereo camera sensor. After updating the 3D probabilistic voxel space with the simulated sensor data, $E[H']$ is calculated similar to H. We select the robot's next exploration goal to be the goal with the highest utility value u_i. We determine u_i for reaching an exploration goal as introduced by [27], taking the distance to the exploration goal d_i and IG_i into account:

$$u_i = IG_i \cdot \exp^{(-\lambda d_i)} \tag{2}$$

We set the constant $\lambda = 0.2$ as proposed by [27]. Depending on the actual distance to the selected exploration goal, the robot either moves towards the goal or looks at the goal. If the exploration goal is close to the current robot position, i. e., within the range of the stereo sensor, the robot looks at the goal: The robot moves its pan/tilt camera head in the direction of the goal and updates the voxel space with the newly received information. If the exploration goal is far away, the robot plans a path to the goal and then tries to approach it.

Similar to [36] and [61], we calculate and evaluate u_i repetitively during the goal approach. If u_i decreases below a threshold δ, the goal approach is stopped. This prevents the robot from moving towards goals leading for example into a dead end, as shown in Fig. 18. After reaching an exploration goal or stopping the goal approach, a new exploration goal is chosen according to the exploration strategy described here.

4.4 Manipulation

To cope with communication delays during remote planetary missions, the manipulation of probes and technical equipment must be semi-autonomous to the least. Round trip times from a few seconds (Moon) up to several minutes (Mars) degrade teleoperation techniques. The manipulation software of the LRU can autonomously pick and assemble prior known objects. An example is the assembly of the

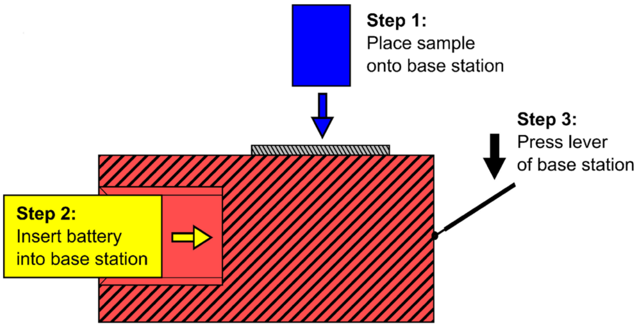
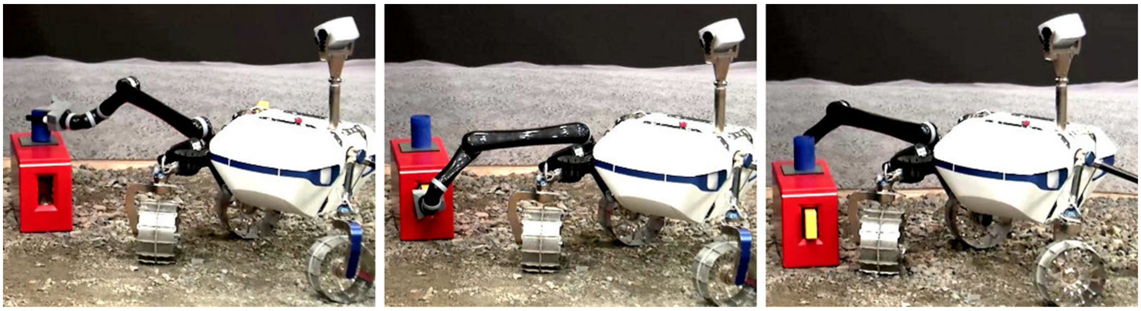

Step 1:
Place sample onto base station
Step 3:
Press lever of base station
Step 2:
Insert battery into base station

Fig. 19 Sketch of assembly tasks performed at the base station object of the SpaceBotCamp



(a) Placing the sample (b) Inserting the battery (c) Pressing the lever

Fig. 20 Our successful base station assembly at the SpaceBotCamp

base station of the SpaceBotCamp, as shown in Figs. 19 and 20. The most challenging subtask is the autonomous insertion of the battery object. The rover has to estimate the relative pose of the base station (see Section 4.1.3), plan and execute a motion to position the battery in front of the slot and finally insert it while actively controlling the contact (see Section 4.5.2). In this section, we describes how the LRU plans motions for its manipulator and combines its manipulation modules in order to solve the individual subtasks.

4.4.1 Manipulation Sequence

After detecting an object, for example the base station, the rover must select a strategy on how to place the sample or insert the battery. We designed a manipulation sequence, which follows a similar pattern for each object. First the rover estimates the object pose, see Section 4.1.3, and approaches the object at a predefined pose. Next the manipulator hand approaches the object at a subsequent predefined pose. Finally the manipulator executes a sequence of relative impedance-controlled motions to fulfill the task. We encoded the individual manipulation strategies as relative transformations in the robot's world model, as described in Section 4.6.3. We manually designed the strategies as our scenario only included known objects and our goal was that each strategy is predictable by the operator. We split the individual approach motions into two parts: First the rover approaches the object based on an initial pose estimation and in a second step the manipulator approaches the object based on a second pose estimation. We choose not to implement a whole-body approach motion in order to minimize the uncertainties in the tool position, as the navigation of the rover is subject to much higher uncertainties (due to slip and sensor noise) then a manipulator motion by its own. In contrast to the relative impedance-controlled motions that we could test beforehand, the manipulator approach motions need to be planned online in order to avoid undesired collisions with the rover itself as well as with the previously unknown environment.

4.4.2 Motion Planning

To allow for any kind of autonomy during manipulation, the rover must be able to plan and execute motions of its manipulator. These motions need to satisfy a multitude of different constraints:

- *Task goal constraints*: The manipulator motion has to translate and rotate the hand or object to the desired goal pose. For example the manipulator has to position the sample above the scale on top of the base object.
- *Continuous task constraints*: The manipulator motion needs to keep the hand or object in a certain position or orientation. For example the manipulator needs to keep the sample container oriented upright in order not to spill its contents.
- *Collision avoidance constraints*: The manipulator motion is not allowed to collide with the rover or the environment, including other objects like for example the base station.
- *Kinematic constraints of the manipulator*: The manipulator motion needs to respect the kinematic structure of the manipulator and its limits.
- *Dynamic constraints of the manipulator*: The manipulator motion needs to respect the dynamic constraints of the manipulator, which is limited by the maximum torques available in its motors. When manipulating heavy objects, the manipulator thus cannot hold the object in arbitrary configurations.

Finding a motion that satisfies all these constraints is not an easy task, due to three main facts:

- *High dimensionality of the search space*: As the Jaco manipulator has six joints, the possible search space has six dimensions. When being discretized with a resolution of 0.1 deg, the unique configuration space has more than 2×10^{21} different configurations, which is roughly half the number of grains of sand on all of the beaches on Earth.

- *Non-linear projection*: The configuration space of the manipulator as well as the workspace it manipulates objects in are connected through the kinematics of the manipulator, which is a non-linear projection. This makes it difficult to perform the inverse projection of collision and manipulation task constraints into the configuration space of the manipulator.
- *Multitude of constraints*: As mentioned above, the manipulator motion is subject to a multitude of constraints, with each on its own already posing a challenge to motion planning. Combining these constraints of different task spaces in a single motion is particularly challenging.

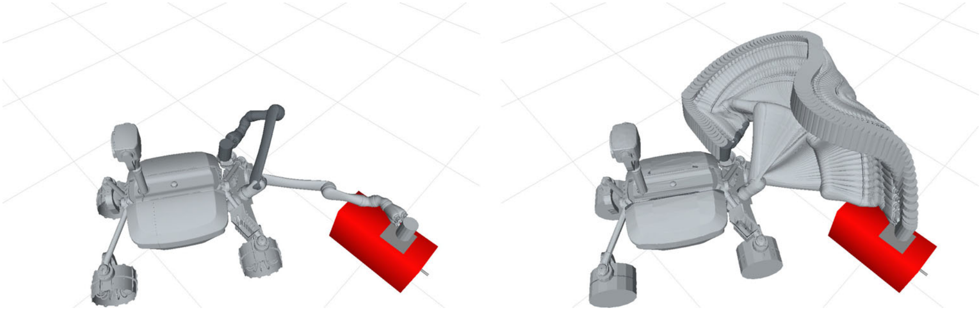
Our approach faces these challenges with a combination of different methods, where each individual method addresses one of the main challenges:

- *Rapidly Expanding Randomized Trees (RRT)*: Our method for manipulator motion planning is based on the concept of RRTs [44]. They face the challenge of high-dimensional search spaces by randomly sampling the search space and dividing it implicitly into Voronoi regions. This biases the search for new motions towards unexplored regions of the search space and reduces the effect of the curse of dimensionality.
- *Linearization of the constraints*: To gather all constraints in one search space, we linearize the projection with the Jacobian of the manipulator and project the individual constraints into the configuration space of the manipulator [62]. By using this linearization in an iterative algorithm, the robot can search the configuration space for possible motions while respecting all constraints.
- *Hierarchical composition of constraints*: To combine the individual task constraints in a defined manner, we prioritize the constraints based on their importance, similar to the concept of [25]. We enforce these priorities by projecting low-priority constraints into the null space of the linearization of high-priority constraints. This allows the robot for example to assign higher importance to avoiding collisions than to holding the sample strictly upright.

For example in the SpaceBotCamp scenario, the LRU has to place the sample onto the ground station. The corresponding approach motion from the rover's transport container needs to hold the sample upright (continuous orientation constraint), reach a pose above the ground station (goal pose constraint), and is not allowed to collide with the rover as well as the ground station (self- and environment- collision constraints). One corresponding motion computed by our motion planner is depicted in Fig. 21. The manipulator motion fulfills all mentioned constraints and moves the sample into the correct position above the ground station. During the SpaceBotCamp scenario, we used the motion planner for adaptive motions from the manipulator's home position to the object-relative approach poses. We however predefined the approach motions themselves to allow more thorough tests of these critical motions beforehand and facilitate the monitoring from our ground station as predefined motions are easier to predict. In future work, for example the upcoming ROBEX mission, we will integrate our motion planner with environment models based on the robot's sensor data and will employ it to plan whole approach motions combining the degrees of freedom from the rover platform as well as its manipulator.

4.5 Control

In this section, we describe the control concepts for both the rover platform and the rover's manipulator as used in the SpaceBotCamp challenge, where locomotion and manipulation have been performed strictly separated. For future challenges, it is desirable to make use of the combined degrees of freedom of rover platform and manipulator. Therefore, the implementation of a suitable whole-body controller



(a) Start (dark) and goal state (light)

(b) Planned solution visualized as a flow of configurations

Fig. 21 A visualization of planning the approach motion from the sample holder to the base station (red). The given start configuration (dark) as well as the found goal configuration (light) are displayed on the left. The computed motion plan, depicted as a flow of configurations on the right, fulfills the orientation constraint of holding the sample upright at all times

exploiting the benefits of compliant control algorithms will be our next step.

4.5.1 Platform Control

The mobile platform of the LRU is equipped with four independently steerable wheels to guarantee high maneuverability. Thus it is able to both turn on the spot and perform car-like as well as crab-like motions according to the situation. Therefore, the wheels have to be steered individually such that the nonholonomic (moving without slipping) constraints are satisfied. A desired velocity in three planar directions (translation in x- and y-direction and rotation about the z-axis) is commanded for the geometric center of the rover from the high-level software components like local path planning, as shown in Fig. 9. This velocity command is then used to compute the corresponding desired velocities and steering angles for the individual wheels by solving the rigid body kinematics equations. We assume that the underlying controllers follow the desired steering positions and wheel velocities sufficiently fast. Experience has shown that this simple control strategy is robust and well-suited for motions in rough terrain, as we demonstrated in the SpaceBotCamp challenge, see Section 6. Nevertheless, slip detection and traction control will be implemented to improve the performance in rough terrain, especially in view of the ROBEX scenario.

4.5.2 Manipulator Control

During the SpaceBotCamp, two known objects had to be localized, picked up, and placed in containers attached to the LRU. Then, the rover had to drive back to the base object, where the container with the sample had to be placed on a smooth surface (scale) on top of the base object, and the battery object had to be pushed into the slot at one side of the base object. Thus, the different requirements during the manipulation scenarios ranged from high positioning accuracy during the approach phase to robust and defined contact behavior during the manipulation phase, especially while inserting the battery object into the slot of the base station. We give a detailed description of the manipulation sequence and the resulting constraints in Section 4.4. In order to satisfy this variety of requirements, we used different control modes during the SpaceBotCamp, including position control and impedance control [13].

To implement the different control modes, we replaced the commercial *Kinova Jaco2 API* with in-house controllers. We developed a communication layer based on an embedded microcontroller, which sends pulse-width modulation (PWM) commands directly to the motor controllers at the necessary control frequencies of approx. 1 kHz and allows the implementation of joint-level position and torque controllers. Building upon the joint torque controllers, we implemented an impedance control framework in the operational space of the manipulator, which is determined by the six Cartesian degrees of freedom of the end-effector. Our impedance control framework is based on the principle of a virtual spring-damper system as sketched out in Fig. 22. The aim is to modify the mechanical impedance of the manipulator, that means the mapping from (generalized) velocities to (generalized) forces [47]. The setpoint of the controller $\boldsymbol{x}_{\text{des}}$ can be understood as the equilibrium point of a virtual spring in the absence of external forces. It is used to compute desired joint torques $\boldsymbol{\tau}_{\text{cmd}}$, which are realized by an inner torque control loop. The resulting robot motion is fed back to the controller. With this approach it is possible to define a proper and stable interaction behavior in contact with both known objects and unknown environments. The inner torque control loop is compliant and thus allows for a good contact behavior for small to medium stiffness of the virtual spring. This approach also leads to a robust task execution, for example when inserting the battery object into the slot of the base station, as the active compliance of the manipulator can be employed to cope with uncertainties in modeling and localization.

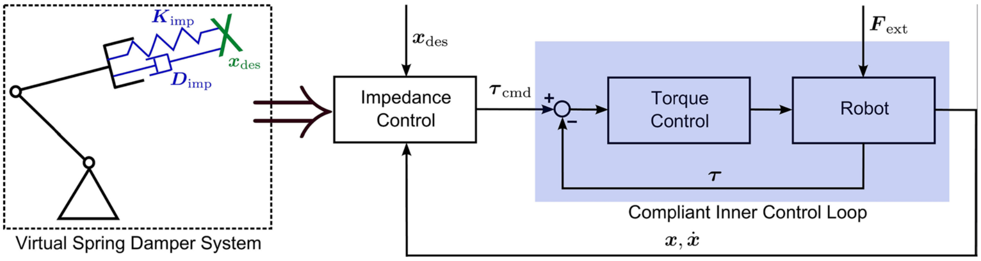


Fig. 22 Impedance control in operational space: The desired position in operational space $\boldsymbol{x}_{\text{des}}$ is converted into joint torque commands $\boldsymbol{\tau}_{\text{cmd}}$ via a virtual spring (stiffness $\boldsymbol{K}_{\text{imp}}$) and a virtual damper (damping constant $\boldsymbol{D}_{\text{imp}}$). The joint torques are realized by a compliant inner control loop under the influence of external forces $\boldsymbol{F}_{\text{ext}}$. Position and velocity of the end-effector $(\boldsymbol{x}, \dot{\boldsymbol{x}})$ are computed by forward kinematics and fed back to the impedance controller

4.6 Autonomous Task Execution

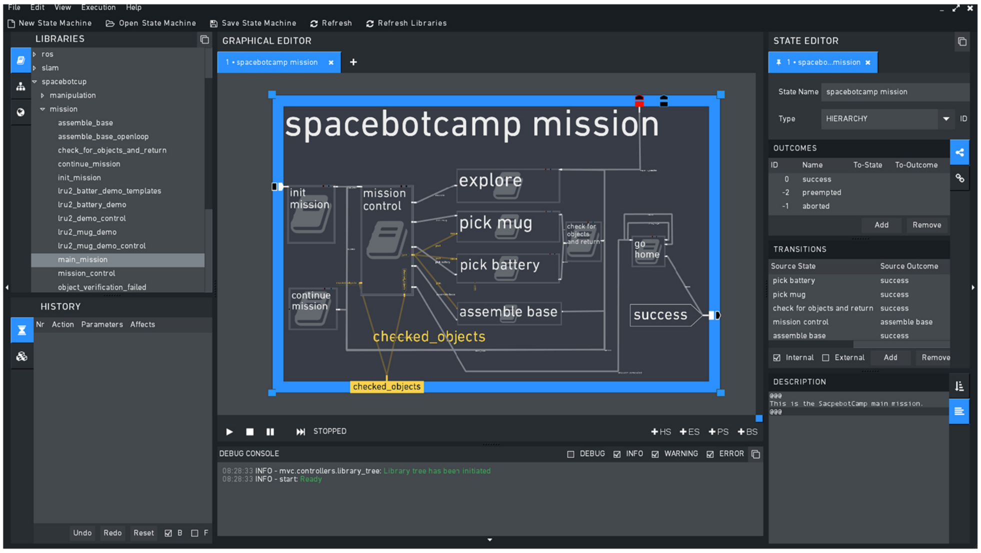
Due to communication limitations, one important requirement for a planetary exploration robot is the ability to fulfill complex tasks without human interaction. To enable a robot to act autonomously, various task control architectures can be found in literature, which can be categoriesed into three approaches: symbolic planners, behavior-based approaches, and congnition-enabled robot control. Symbolic planners are the most commonly used architectures to solve complex scheduling tasks. The task is defined as a target world state and the planner tries to compute a sequence of actions to transform the current state into the desired one. Thus, the state of the environment has to be known and the effects of actions on it have to be precipitable. Planetary exploration in unknown environments, by definition, violates at least the first of these two assumptions. Because of these limitations, Brooks [18] proposed an architecture based on a hierarchy of simple behaviors. While it was shown to work well in real-world environments for tasks like avoiding collisions and random exploring, this approach is not sufficient for more complex tasks like the maintenance of technical equipment. The cognition-enabled robot control [15] is bridging these contradictory approaches by proposing a reactive behavior specification while maintaining a semantic representation of the behaviors. It was designed for domains, in which the tasks are strongly underspecified, like service robots doing home chores. Therefore, the inference of missing information and its transformation into executable plans are important aspects. In planetary exploration, the task itself however is precisely defined. The challenge for the robotic system is to fulfill it robustly in an unknown environment with the uncertainties created by real hardware and software components. For our architecture, we therefore propose an approach that is similar to the cognition-enabled robot control with respect to the reactive behavior of the control by using plans, but, since our tasks are better specified, we can use fixed implementations of the plans themselves. This allows us to test each sub-plan in detail, making the system more robust and also predictable for a human operator.

In the following sections, we introduce *RMC Advanced Flow Control (RAFCON)* as our tool for implementing autonomous task execution. We present our concepts on how to implement the plans and how to model the robot's world state. Furthermore, we utilize the SpaceBotCamp scenario as an example for the mission design and highlight the advantages of our approach w. r. t. fault detection and recovery.

4.6.1 RAFCON

We employ our own powerful visual programming tool RAFCON [19, 20] for programming complex autonomous tasks. It is based on hierarchical, concurrent state machines. For the SpaceBotCamp challenge, we built state machines with more than 700 states, 1200 transitions and up to eight hierarchy levels, demonstrating the capability and scalability of the tool. The handling of huge state machines is possible due to the state machine editor GUI, see Fig. 23, featuring an elaborate zooming concept. This allows the user to zoom and pan inside the state machine like in a digital map and thereby facilitates dynamic expansion or hiding of details of sub-states deeply nested in a hierarchy.

Moreover, RAFCON supports state re-usability as state machines, designed by different developers, can be linked



- ros
- slam
- spacebotcup
  - manipulation
  - mission
    - assemble_base
    - assemble_base_openloop
    - check_for_objects_and_return
    - continue_mission
    - init_mission
    - lru2_batter_demo_templates
    - lru2_battery_demo
    - lru2_demo_control
    - lru2_mug_demo
    - lru2_mug_demo_control
    - main_mission
    - mission_control
    - object_verification_failed

State Name: spacebotcamp mission

Type: HIERARCHY

OUTCOMES

ID	Name	To-State	To-Outcome
0	success		
-2	preempted		
-1	aborted		

TRANSITIONS

Source State	Source Outcome
pick battery	success
pick mug	success
check for objects and return	success
mission control	assemble base
assemble base	success

☑ Internal □ External

DESCRIPTION

```
@@@
This is the SacpebotCamp main mission.
@@@
```

```
08:28:33 INFO - mvc.controllers.library_tree: Library tree has been initiated
08:28:33 INFO - start: Ready
```

Fig. 23 The RAFCON GUI, showing the state machine that we used at the SpacebotCamp in its central widget

into more complex state machines by so-called library states. In scenarios like the SpacebotCamp challenge, state machine development is typically distributed among several developers. In our case, one cared for the states concerning computer vision, another prepared the navigation states and a third wrapped the manipulation functionalities of the robot into states. Thereby, we can provide all of the robot's functionalities in small state machines that we then combine to more complex ones. Furthermore, these single building blocks can be re-used in other scenarios, reducing development time. A human-readable and thus versionable state machine storage format improves collaborative state machine creation. Next to linking existing states as libraries, they can also be used as templates to quickly develop similar functionality.

The ability to launch state machines at arbitrary states boosts quick state machine development and makes powerful error handling and failure recovery possible. Especially in complex missions, in which a part of the task has already been fulfilled, dedicated entry points allow a full system recovery without performing redundant work. An open plugin concept offers an easy integration of scenario-specific data visualization. For the SpacebotCamp, we developed a plugin for remote state machine execution monitoring and control, as described in Section 4.7.

4.6.2 Plan Representation

A plan is defined as a control program that encodes a strategy for the robot on how to complete a task [15]. As in the planetary exploration domain, typically all tasks are well defined, there is no need for an automatic generation of plan segments. Instead, we propose a modular concept of hierarchical plans that have been designed beforehand and can thus be tested in the preparation phase of a mission. Even though we do not use any plan generation, the robot's course of action is not fixed, but depends on its sensor data and the knowledge it acquires during plan execution. In order to create efficient and flexible plans that scale to more complex problem instances, we set up several design rules, which we describe in the following. For the implementation of these rules, we strongly benefit from the features of our tool RAFCON.

- *Modularity: A plan is required for each identifiable sub-task*: Breaking a mission down into the smallest possible sub-tasks with respective plans is beneficial for various reasons. First, code duplication can be avoided by reuse of already implemented plans, exploiting the library concept of RAFCON states. Second, the task's structure and, during execution, the mission state are better observable in RAFCON's visualization when all individual sub-tasks, ideally down to each functional call, are represented as separate (hierarchical) states in the control program. Third, it is easier to detect causes of failures when the individual units are smaller. Fourth, there is less hidden data flow between different function calls when they are represented as RAFCON states, since data flow can be visualized and monitored in our tool.
- *Robustness: A plan should recover locally*: In cases where function calls fail to cope with uncertainties of the environment, it is important to detect failures and try to recover locally. For example, a nondeterministic heuristic may find a solution simply by trying again. For most errors however, more sophisticated fallback strategies are necessary. In RAFCON, they can be added as additional states that are executed on the detection of failures in individual states. If no local error recovery is possible, the RAFCON error propagation passes the problem to a higher level in the state hierarchy.
- *Generality: A plan should have no dependencies*: The design goal of a plan is that it can be executed whenever the robot has to fulfill the corresponding task. Thus it should not depend directly on previously executed plans. Of course, this is not possible if the task itself depends on previous tasks. Nevertheless, this decoupling should be the design goal for each plan. An example on how to decouple plans is to enforce that each plan using the robot's manipulator drives it back to its home position. This allows all plans to start with the assumption of a known initial manipulator pose. Introducing such constraints might slightly reduce efficiency but make the plans much more flexible.

4.6.3 World Representation

Task execution approaches based on symbolic planning require a detailed model of the world state for planning. As mentioned before, this however is unavailable in unknown environments. In contrast, our approach does not employ the world representation for direct planning. Instead, it determines the robot's course of action by selecting appropriate pre-defined plans that can cope with uncertainties, unexpected events, or failures, as discussed in Section 4.6.2. Since we do not employ our world representation as the basis for planning, we avoid problems arising from oversimplifications. We use it to model the following knowledge of the robot:

- *Environment properties*: Even though the environment is not fully known, we typically do have significant initial knowledge a priori to a mission. One example is information about technical devices that we want to manipulate. Such knowledge has to be represented in a way that is useful for the robot. We employ a scene-graph model to represent the world as a tree of nodes that are linked by 6D transformations to their respective parents. Each node represents either a physical object or an abstract entity like a pose, a grasp or a region of interest. Annotated links between these nodes allow us to also model uncertainties, incomplete and missing information. For example, while the initial location of a certain object might be unknown, we can model its existence as a-priori knowledge by adding it as a child of the world node. In addition, we can include further information as its properties and child nodes, like for example possible grasps defined relative to the object. Once we detect and localize the object in the environment, we can easily update our world model by re-locating the object's sub-tree in the scene graph.

- *Semantic interpretation*: The decisions regarding the robot's task execution within our hierarchical plans are based on the semantic interpretation of explicit and implicit knowledge represented in the world state. It thus needs to have a task-related meaning for the robot. Some facts about the state of the environment, the robot and its mission need to be represented explicitly, for example if a button on a technical device has already been pressed or not. Other knowledge can be derived from the current state of the scene graph. During pick and place tasks, an object can, for example, be inferred to be successfully picked up by the robot once its parent node in the scene graph is asserted to be the robot itself.
- *Memory of experience*: Not every piece of information can be immediately employed within the (sub-)plan in which it is acquired. An example from the SpaceBotCamp would be the detection of a target object while executing the pickup of another. It is important to store such information in the robot's world representation for later use, when deciding on the robot's next actions. In addition, storing experience as knowledge about past failures allows to prevent their repetition and thereby protects the robot from getting stuck in oscillating behavior. For example, consider a case in which the robot, from far away, confuses a stone with a target object to be picked up. When approaching the object, at closer distance, it is able to resolve its error. Without memory however, the robot might drive away and repeat its mistake from a further distance, approaching the same stone over and over again. Storing information in the world model stating that the target object is not at the stone's location can prevent such issues.

4.6.4 Mission Control

To deal with the limited communication that is typical for space missions, we propose to increase the autonomy of the system. This implies that the mission control itself is to a certain extend located on-board the robot. As long as no intervention from the ground station occurs, the system has to decide on its course of action on its own. For high-level plans, especially the top level, we developed a clear structure to implement this behavior with our tool RAFCON. In Fig. 24, we present a simplified version of our top-level state machine for the SpacebotCamp mission. Besides the two initialization states, we created several sub-plan states, like *pick battery*, *assemble base*, etc., and a central decision state called *mission control*. This *mission control* state decides which sub-plan is executed next based on the current world state and a set of rules. For example, if a battery object is detected and no battery has already been picked up, the *pick battery* plan will be executed. Afterwards, the *mission control* state will be activated to decide for the next step, which can be dependent on the success or failure of the previously executed sub-plans. Therefore, the robot's course of action is not fixed but results from the robot's current knowledge about the world and the task structure, which is encoded as a set of rules. This structure allows for a flexible control flow that is easy to understand and monitor. In case failures cannot be resolved locally, knowledge about them is added to the world state. It can thus be taken into account for future autonomous decision making without the need for changing the state machines themselves.

Based on this structure, we can also simplify the interactions with human operators at the ground station, as described in Section 4.7. Operators can easily pause, stop and re-start the state machine as they do not have to reason about the correct entry point. The robot can determine it by

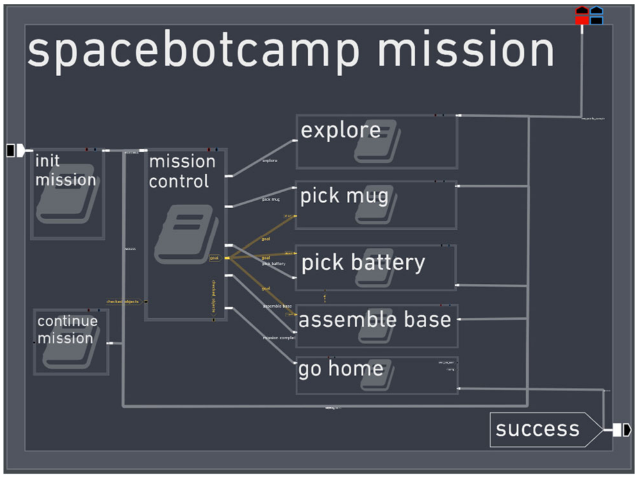


Fig. 24 Simplified SpaceBotCamp top-level state machine containing a single decision state, *mission control*, that decides on which sub-plan to execute next based on the current world state and a set of rules. This structure allows for a flexible course of action and simplifies error handling and human interaction. The *pick mug* state depicted in this screenshot refers to the state for picking up the rock sample

itself based on the knowledge represented in its world state. Since the world state itself is saved in a persistent way, even a complete reboot of the robot is unproblematic hence the state machine can automatically continue with the mission afterwards.

4.7 Ground Station Mission Control

Being able to monitor and – if necessary – to intervene with a robot's task execution is crucial while operating a robot in unstructured and unknown environments. In contrast to many field robotics applications here on Earth, the communication channel to a robot deployed on another planet will likely be low-bandwidth, severely delayed and unreliable with potentially extended periods of communication blackouts in one or both directions. We designed our system and mission control to efficiently monitor autonomous task execution as well as to control the robot with periods of shared autonomy when necessary.

4.7.1 Communication Setup

We present a sketch of our communication setup in Fig. 25. The autonomous LRU rover is connected via a wireless connection to an on-site lander, as depicted in the left part of the figure. The only responsibility of the lander is to forward the communication link over the delayed and unreliable channel between the robot system and our ground station. All processes controlling the autonomous robot operation run on the LRU rover itself. The ground station (mission control), as shown in the right part of the figure, is split into a main computer that acts as the communication counterpart of the lander. In addition, it runs the server processes for the ground station network in order to distribute the data among several operator workstations for remote monitoring and access.

4.7.2 Dealing with Unreliable Communication Channels

To avoid any side-effects of an unreliable communication channel on the robot's task execution, all communication between the robot and human operators at the ground station is based on non-acknowledged communication over UDP between otherwise completely decoupled networks. Not using any acknowledgment mechanism for communication fits well to many transmitted data streams of continuous nature (e. g., sensor data) and allows the robot to operate in scenarios where no uplink to the robot is available (the default in the SpaceBotCamp scenario).

For transmitting task-relevant information like sensor data, planned paths or detected objects, we use ROS in a multi-master setup with two ROS networks, one on the robot and one between all ground station computers. We developed an UDP forwarder to exchange any ROS topics in a network-transparent way between subscribers and publishers on both sides. High-bandwidth data like images and point clouds are transferred rate-limited and with reduced resolution. For specific ROS message types like images, the data is split and marshalled in a way that allows to reassemble a complete ROS message even if large parts of the message's packets have not been received. We, for example, transmit image data chunk-wise and fill in missing chunks on the receiver side.

We monitor the execution status of ROS and non-ROS processes running on the robot through our process manager provided by our *Links and Nodes* middleware, thereby gaining access to all processes' console output. In addition, we get a clear picture if all required components operate normally by specifying run and restart dependencies as well as conditions for how a process normally behaves during start-up, execution and quitting (defined by regular expressions on console output). The console outputs and execution state (*stopped*, *starting*, *started*, *ready*, *stopping*, *error*, etc.)

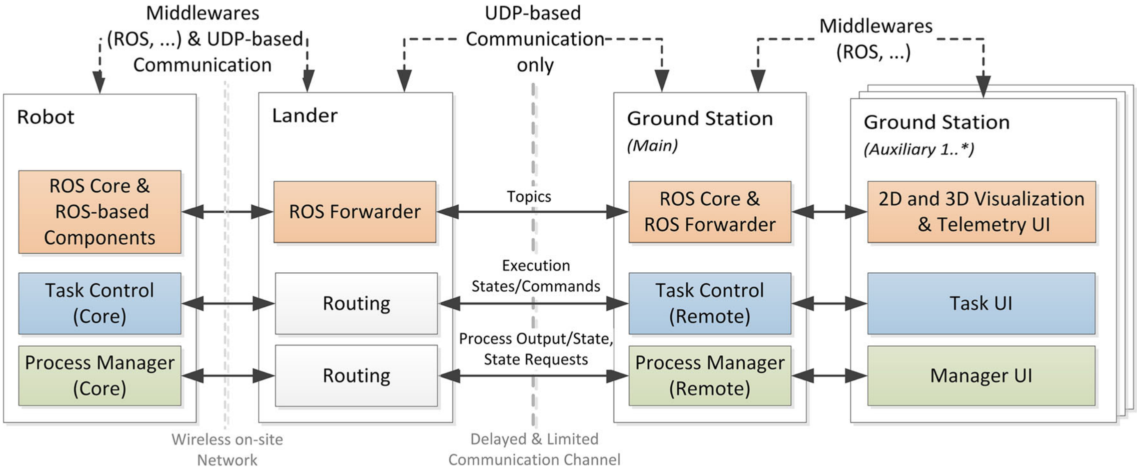

Middlewares (ROS, ...) & UDP-based Communication
UDP-based Communication only
Middlewares (ROS, ...)
Robot
ROS Core & ROS-based Components
Task Control (Core)
Process Manager (Core)
Lander
ROS Forwarder
Routing
Routing
Topics
Execution States/Commands
Process Output/State, State Requests
Ground Station (Main)
ROS Core & ROS Forwarder
Task Control (Remote)
Process Manager (Remote)
Ground Station (Auxiliary 1..*)
2D and 3D Visualization & Telemetry UI
Task UI
Manager UI
Wireless on-site Network
Delayed & Limited Communication Channel

Fig. 25 Communication setup for remote monitoring and access

per process are forwarded to the ground station via UDP multicasts, allowing multiple operators and applications to receive and analyze this information at the same time. The process manager on the robot side also implements a UDP command protocol over which start/stop requests for any process can be triggered from the ground station. To reliably monitor the task execution and high-level system state, our task control software implements a UDP-based communication protocol (separate from our mechanism for ROS messages) with additional message bursts, sequence numbers and hash values to cope with packet loss and reordering during transmission of state information. Similar to the process manager, a UDP-based command protocol allows, for example, to pause and continue task execution.

4.7.3 Remote Access for Shared Autonomy

In case of (imminent) failure or on the observation of abnormal behavior on system- or component-level, we employ several mechanisms to restore normal operation conditions and resume autonomous task execution. On a task-level, we are able to start, stop, pause and step through the active task execution model. In addition to the main task, we maintain a repository of standalone (sub-)tasks, which can be triggered remotely and largely require no specific start conditions (e. g., *lookBackwards*, *moveManipulatorToHomePosition*, *detectObjectA*). On a process level, we are able to stop and restart processes and process groups on the robot. Besides those required to execute a task with the robot, we also prepared failure handling scripts, which can be triggered for example to re-initialize hardware components or modify parameters. For remote terminal and direct file-system access, we employed the mobile shell Mosh [10], which is available as free software, in order to replace the usual SSH terminal connection.

4.7.4 Mission Control Setup at Ground Station

Our mission control setup at the ground station, as used during the SpaceBotCamp challenge, includes three operators – *navigation*, (general) *system* and *manipulation operator* – and one supervising *operator coordinator*. The navigation operator monitors planning and execution of drive motions as well as robot localization and mapping through appropriate 2D and 3D visualizations for navigation frames, paths and the generated maps (obstacle map, height map, etc.). The system operator focuses on warning/error monitoring for all processes, (sub-)system restarts, monitoring and, if necessary, control of high-level task execution. For this purpose, we use the ROS console together with custom remote UIs for process management and high-level task visualization/execution. The manipulation operator monitors the object recognition, manipulator motion planning and motion execution. All operators have access to all basic robot telemetry information such as battery voltage, CPU usage and temperature, emergency status, joint positions & torques, controller state, and control rate statistics via a custom UI as well.

5 Simulation Environment

We built a complete Software-in-the-Loop (SiL) simulation [45] that replicates all the interfaces of the LRU, including its drives, IMU and visual sensors. Hence, the software components may run on both the real and the simulated system without modifications. The simulation allows parallel software development despite limited hardware resources and pre-testing of software components in different scenarios. The latter is particularly important for space missions as real-world tests are difficult for extraterrestrial environments.

5.1 High-Fidelity Robot Simulator

The *RoverSimulationToolkit* [31] is a general purpose toolkit for the simulation of planetary rovers, written in Modelica and currently developed at DLR. It focuses on multi-body simulations and in particular on the locomotion subsystem to support the design and development of new rovers during the early design phases. It allows the user to quickly design new rovers, modify existing designs, model a vast variety of environments and to test the rovers in those environments. In preparation for the SpaceBotCamp challenge, we employed this toolkit to create a SiL simulator for the LRU. It replaces the sensors and actuators, i. e., the first and last row of components in the block diagram in Fig. 9, with a simulation, while leaving all other parts as they are. In Fig. 26, on the right, we show the rover's software stack, while on the left side, we present what it communicates with: Either the rover in the real physical world or a simulation thereof, including both the simulated environment as well as the rover with its virtual sensors and actuators. We grayed out the manipulation components as their simulated counterparts are current work in progress and not fully integrated yet.

When simulating a rover with our *RoverSimulationToolkit*, we automatically create an interactive 3D visualization utilizing the *DLR Visualization Library* [31]. As we described in Section 4.2, the LRU's autonomy is primarily based on optical sensors. Therefore simulating the camera system became an essential part of our SiL simulator: We utilize the 3D visualization to create artificial camera images, which then can be used by our computer vision algorithms [30, 54]. Therefore we extended the simulation to provide renderings of camera images with a predefined frame rate and independently from the user visualization.

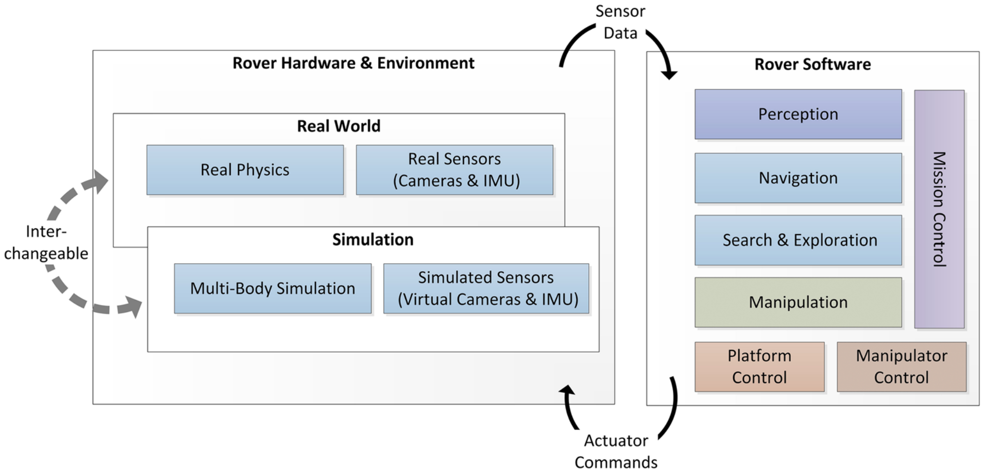


Fig. 26 Software-in-the-Loop (SiL) simulation: The rover software sends actuator commands to either the simulation or to the real rover and receives sensor data. The simulation is build in such a way that the switch between the two systems does not require any changes to the rover's software components

The resulting color or monochrome images are distributed via our *SensorNet* middleware, equally to the interfaces on the real rover system. The usage of multi-threading and modern CPU extensions [63] allows us to render multiple cameras, each running at a rate of 15 - 30 Hz. The large bandwidth when handling virtual camera data does not only affect image creation, but also the image processing pipeline. As we described in Section 4.1, the stereo matching for 3D reconstruction is a computationally expensive task, requiring the use of FPGAs to run sufficiently fast. While we can interface our FPGA-based stereo reconstruction from the simulation, the number of FPGA boards however is limited. To circumvent this issue, we further extended our simulated cameras with the option to directly provide simulated depth data. While these artificial depth images differ from real stereo reconstructions, in particular w. r. t. error and noise behavior, they are sufficient for many steps of unit and integration testing. In Fig. 27, we show the different parts of our 3D visualization. The left image shows a visualization as it presents itself to the user: A 3D visualization of the scenario, in this case the LRU driving on the surface of the Moon. The other images show virtual camera views: The center one shows a normal camera image and the right one a depth image of the same scene, with colors of each pixel denoting the distance from the camera. The color gradient indicates the receding ground and the large boulder in front of the rover.

5.2 Simulation Scenarios and Tests

In preparation for the 2015 SpaceBotCamp, we primarily used an environment derived from the previous challenge in 2013, which was recorded by the team of the University of Bonn, Germany. They publicly released a 3D model [37, 53] based on high-precision laser scan data. We simplified their mesh to speed up the simulation and added a surface texture to increase the realsim for our computer vision pipeline. In Fig. 28, we present the LRU in this environment, as well as the corresponding 3D reconstruction created by the LRU's image processing and mapping components.

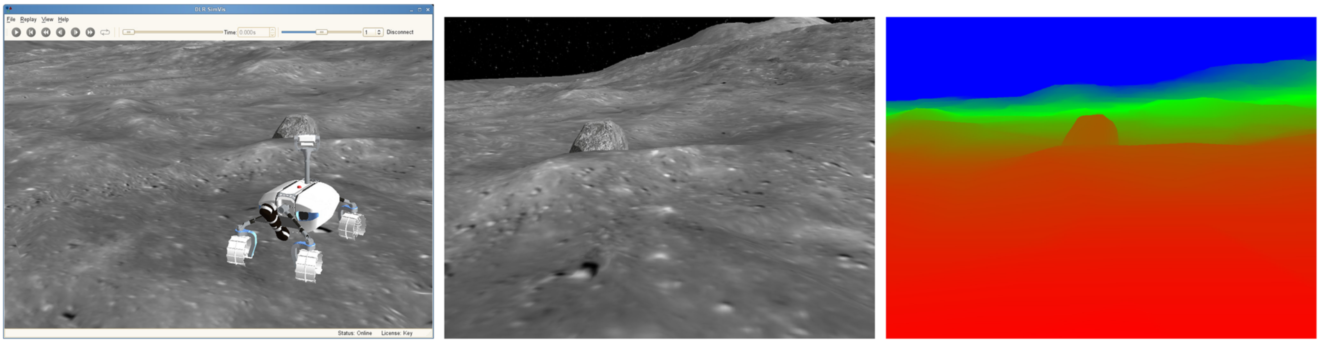


Fig. 27 Simulatior visualization of the LRU on the Moon (left) with virtual camera image (center) and virtual depth camera image (right). The color denotes distance from camera, its gradient indicates the receding ground with the boulder sticking out

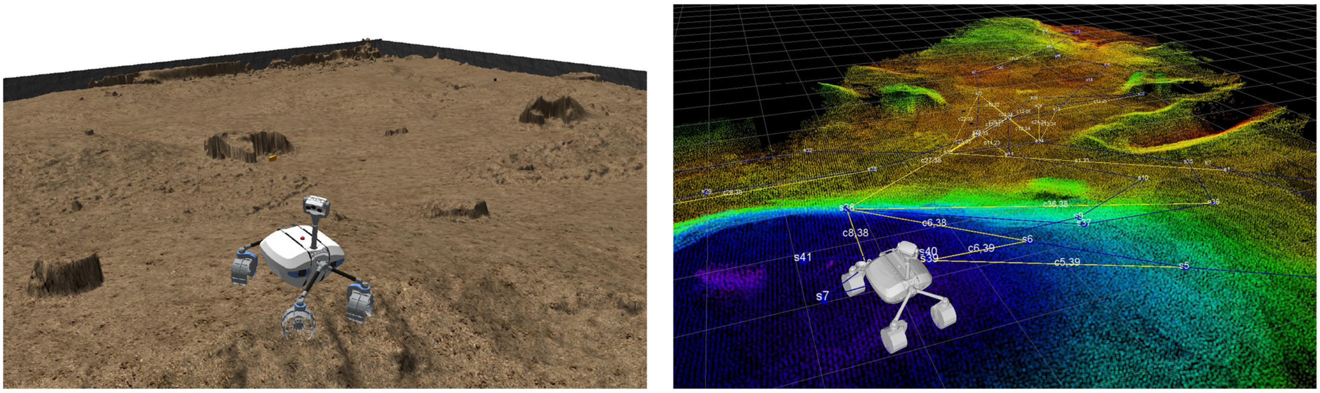


Fig. 28 Landscape of the 2013 SpaceBotCup scenario in the simulator visualization (left) and respective 3D point cloud created by our SLAM system (right)

The upcoming space analogue mission of the ROBEX project will be at Mt. Etna in Sicily, Italy, as we discuss in detail in Section 7.2. Tests in simulations will again play a crucial part in our preparations. Therefore, we recorded a first set of laser scans and panoramic images of a similar, Moon-like environment during a preliminary expedition on the island of Vulcano, Italy in the summer of 2015 [12]. After meshing and simplifying the laser scans, we used the camera image as a background image as well as to create a texture for the 3D model. The result is a realistic reconstruction of the recorded site as depicted in Fig. 29, demonstrating the viability of this approach. Since then, the final mission site has been determined and a detailed exploration of the site on Mt. Etna was performed in summer of 2016. Part of this expedition was the recording of high-resolution images and laser scans of the mission site. Based on this data, we will create a realistic simulation in preparation of the upcoming lunar analogue mission [5].

The ultimate goal for the research of planetary rovers is the deployment on another planet. While for our prototype, this is still far off, we can already use simulations to investigate the behavior of our rover under various conditions present at its final deployment location early on in the design process. Unfortunately, high resolution images that include elevation data of the surface of the Moon are hard to come by. Typically, the highest resolution obtainable is in the order of 50 cm × 50 cm per pixel [7, 41, 68]. This is too low for any meaningful simulation. Yet the lunar surface shows a self-similarity where large features resemble smaller ones [22]. We therefore used images from the *Lunar Reconnaissance Orbiter Camera* at [7] with a resolution of 50 cm × 50 cm per pixel and reduced their metric scale by a factor of ten. Afterwards, we manually added some additional features relevant for navigation, such as rocks. While we cannot guarantee its realism, this virtual Moon-like terrain optically resembles the surface of the Moon, as can be seen in Fig. 27. It therefore allows us to test our image processing pipeline and higher-level components under Moon-like conditions.

Our high-fidelity simulation allows us to test a variety of system components, as indicated in Fig. 26. So far, we primarily employed the simulator to test our stereo vision-

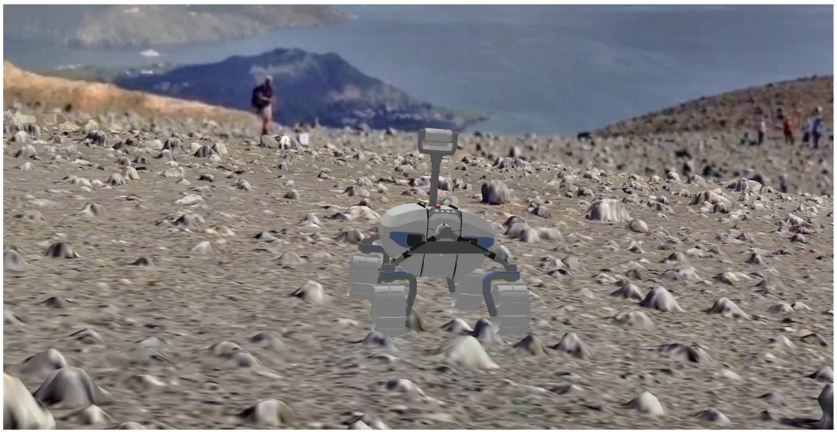


Fig. 29 Simulation of the LRU at a hillside on Vulcano, Italy. Ground reconstructed based on laser scans and panoramic images

based perception pipeline, our localization and mapping components as well as our modules for autonomous exploration and high-level task execution. Future work constitute simulator-based tests of object manipulation tasks, requiring an extension of the simulated rover to include the manipulator. Due to trade-offs between runtime and precision, simplified models and approximations as well as different types of sensor noise, a simulation typically cannot replace tests with a real robot, in particular for low-level components like robot control and raw sensor data processing. Nonetheless, the simulation allows us to test many components and their interplay early on and with less effort compared to real-world tests. This is in particular invaluable to support a continuous system integration, both for initial interface and communication tests between our more than 100 software components as well as for regular integration steps to check if any changes on individual modules influence or break the overall system behavior.

6 The SpaceBotCamp Challenge

The SpaceBotCamp [2, 3] is a national German robotics challenge organized by the DLR Space Administration. Similar to the *DARPA Robotics Challenges (DRC)* [4] in the USA, its goal is to stimulate innovations, benchmark state-of-the-art technologies in the field of autonomous mobile robotics and kick-start their integration into working systems. It took place in November 2015 in Hürth, Germany with ten participating teams from universities and research institutes from all over Germany. We all had to face a challenging task focused on autonomous exploration and manipulation in an unstructured, previously unknown GPS-denied environment.

6.1 Scenario Description

An autonomous robot system, consisting of one or multiple robots with a total mass of less than 100 kg, had to autonomously explore and map an area modeled after a Moon-like, rough-terrain planetary surface. Therein, it had to locate and collect both a blue container with a rock sample (approx. 500 g) as well as a yellow object representing a battery (approx. 800 g). Both objects had to be transported to a red base station and finally assembled, as sketched out in Fig. 19. A high level of autonomy was necessary to fulfill the task since an artificial delay of four seconds round trip time was additionally added to the communication link in order to simulate the real delay between Earth and Moon. The bandwidth was limited to 100 Mbit/s, a data rate that has been reached via Ka-band radio for lunar-Earth distances by the *Lunar Reconnaissance Orbiter* launched in 2009 [39]. Furthermore, the uplink to the robotic system was completely blocked, except for up to three five-minute checkpoints. During these limited time frames, a ground station crew was allowed to send commands to the robotic system over a channel with the same delay and bandwidth limitation as the downlink. Apart from that, the teams could only passively monitor the robot. In addition, the ground station crew was located in a separate room and thus without any visual contact to the competition field. After two days of preparation, each team had a single opportunity to solve the challenge within a sixty-minute time slot in front of a public audience.

6.2 Results and Discussion

According to the original rules of the challenge, we were the only team amongst the ten competitors to fulfill all mandatory tasks. In addition, our robot was able to localize itself in the environment solely relying on its on-board sensors and created a 3D map of the environment. Furthermore, we solved the tasks while facing all of the specified communication constraints from the very beginning of the mission. We accomplished this in just thirty minutes, half of the given time frame, and with full on-board autonomy. In contrast, many other teams softened the challenging communication restrictions in order to allow for mixed autonomy and teleoperation approaches. We only took a single one of the three allowed checkpoints to double-check the object localization for the base station, as its precision is crucial for flawless assembly. We thereby solely sent four high-level commands to the rover during the whole mission while experiencing the one-way delay of two seconds for all sent commands and received data. Thus we could demonstrate that we still had full high-level control of the system despite the delayed and constrained communication link between ground station and LRU. As we perform all processing on-board the LRU, the bandwidth-limited communication channel was sufficient to monitor the system. In particular for the camera images and navigation maps, we down-sampled the data both in frequency and resolution before sending them to the ground station. Our methods described in Section 4.7.2 further allowed us to deal with packet loss that occurred when approaching the limits of the unidirectional communication channel without acknowledgment mechanisms. Finally, our system was the only robot which managed to climb and descend the steep crushed-stone ramp, shown in Fig. 30, fully autonomously. We describe the details of our solution for the SpaceBotCamp challenge throughout this work together with the respective individual components of our system, highlighting all aspects relevant for the SpaceBotCamp as well as the improvements and extensions we made since then. In addition, we present our run at the SpaceBotCamp in the accompanying video (Supplementary Material) and at https://youtu.be/wCTkSxcna8o.

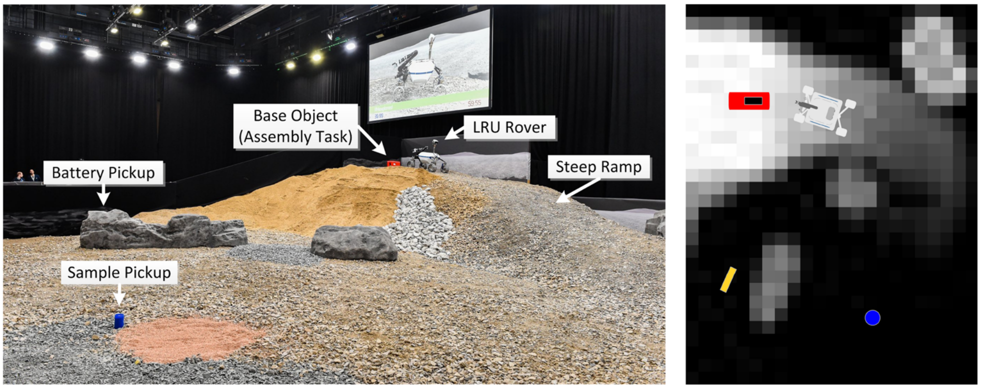


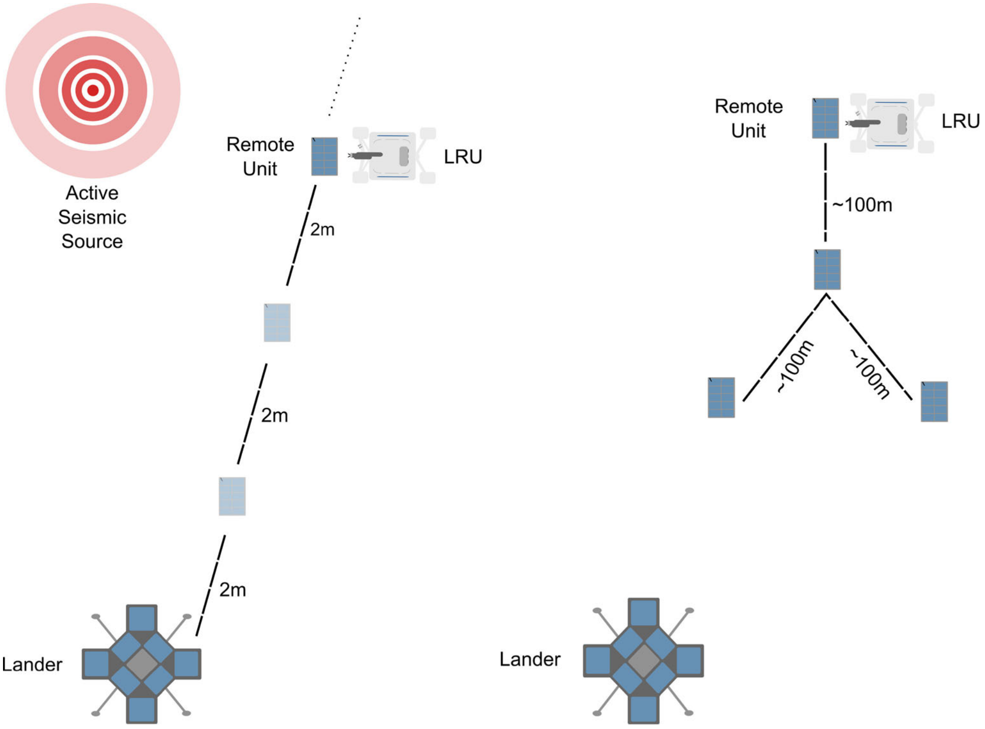
Fig. 30 An overview photo and top-down view of the SpaceBotCamp scenario. The competition field was 13 m x 18 m. The top-down view is underlaid with the given rough height map of the scenario with a resolution of 1 px = 0.5 m. We scaled the sketched LRU and objects up by factors two and four respectively for better visibility. The LRU started next to the red base station on top of a hill of approx. 2 m height. In the foreground, a blue container filled with a rock sample can be seen, while the yellow battery object is hidden by a large rock in the left image. The goal of the challenge was to explore and map the unknown terrain as well as to locate and collect the sample and the battery object for the assembly task at the base station

7 The ROBEX Space Analogue Mission

After the success of our LRU rover in the SpaceBotCamp challenge, we continued to further improve our system, moving in the direction of space missions. Our next milestone is a lunar exploration analogue mission as part of the ROBEX project.

7.1 The ROBEX Alliance

The Helmholtz Alliance *Robotic Exploration of Extreme Environments (ROBEX)* is a conglomeration of sixteen institutes and universities that are performing space and underwater research in extreme environments [1]. This alliance works together to develop advanced robotic technologies



(a) Active seismic experiment

(b) Passive seismic experiment

Fig. 31 Demo scenarios of the ROBEX lunar analogue exploration mission: The LRU has to autonomously place remote units with seismic sensors in two different configurations

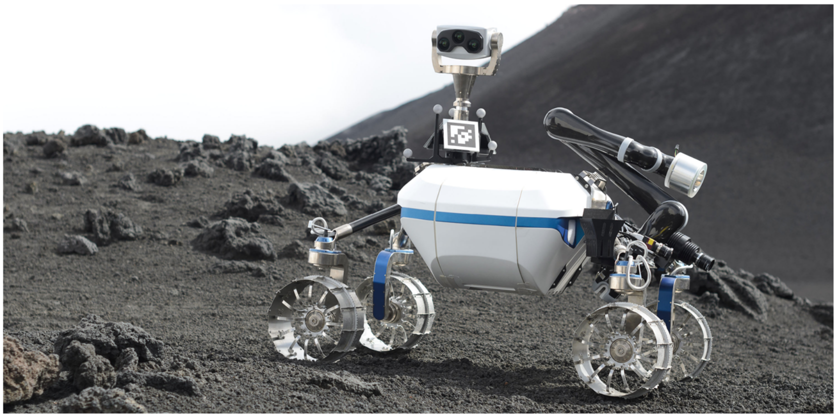


Fig. 32 LRU at lunar analogue test site on Mt. Etna, Sicily, Italy in September 2016

for exploration in harsh environments like deep sea and space. The main objective of this alliance is to bring together deep sea and space research communities, identify common areas of knowledge and technologies that are shared between them, and to learn and benefit from each other. In order to evaluate as well as to showcase the output of the combined efforts of the ROBEX partners, two demo missions are planned at the end of the project: One for deep sea exploration and the other for planetary surface exploration. The planetary exploration scenario relies on a lunar mission scenario and will be demonstrated in an analogue mission on Earth. The mission goal is to study the crust model of Earth's moon based on seismic measurements. The necessary measurement instruments are to be deployed by a lunar rover at different locations around a lander site. Therefore, in the analogue mission, our rover prototype, the LRU, will autonomously deploy a passive seismic array of instruments as well as perform an active seismic measurement campaign [23].

7.2 The Lunar Exploration Analogue Mission

The lunar surface exploration analogue mission is scheduled to take place during June - July 2017. The purpose of the mission is to show the technical readiness of the system in achieving the scientific challenges in a terrestrial scenario, as an actual mission on the Moon would be out of the scope of this project.

For this mission, the selected site should be analogous to the Moon's surface and in addition allow real seismic experiments, as required in the scientific mission scenario. Mt. Etna on Sicily, Italy was selected as the site for the analogue mission, as it is of volcanic nature, features a Moon-like surface and has constant seismic activity. The mission, i. e., ROBEX-ASN as shown in Fig. 31, consists of two parts, which are an active seismic experiment and a passive seismic experiment.

The active experiment will feature an artificial seismic source at a known location. The LRU will pick up a remote unit, i. e., a seismic sensor box, from the lander and carry it to a pre-defined position. There it will places it on flat ground, detach from the box and wait until a seismic measurement is taken. The LRU will then repeat this process at a given set of measurement points, as we sketched out in Fig. 31a. For the passive seismic experiment, a Y-shaped array of seismic sensors is to be deployed as shown in Fig. 31b. In contrast to the active seismic experiment, the passive passive array will record data from the volcano's natural seismic activity. Similar to the SpaceBotCamp challenge, a high level of autonomy for our rover is key to a successful and efficient execution of the mission. Both experiments will be controlled and monitored from a ground station in approx. 30 km distance at the foot of the mountain. We conducted preliminary experiments on Mt. Etna in September 2016 to test the performance of the key components needed for the success of the analogue mission, including the components for autonomous navigation, mapping and manipulation as well as the communication setup to the far-away ground station. In Fig. 32, we give an impression of our LRU operating in the lunar analogue test site on Mt. Etna. Further, we obtained laser scan and image data of our target environment in order to reconstruct the scenario for tests in simulation, as described in detail in Section 5.2.

8 Conclusion and Future Work

In this work, we have presented the Lightweight Rover Unit (LRU) as an agile rover system for autonomous planetary exploration. We provide an overview of our system architecture, detailing both the hardware and software components as well as our ground station setup for monitoring the robot's state and activities over a delayed and restricted communication link. We designed the LRU to operate at a high level of autonomy during rough-terrain navigation, search and exploration as well as object manipulation tasks. This is essential to conduct efficient space missions in the light of delayed communications to foreign planets. In addition, our framework for autonomous task execution allows interactions with the rover at a high level of abstraction in case shared autonomy is needed. The LRU faced the challenges posed by the SpaceBotCamp 2015 planetary exploration scenario with great success and allowed us to demonstrate fully autonomous operation with very limited options for monitoring and remote access. The upcoming ROBEX analogue mission will give us the opportunity to demonstrate the LRU's capabilities to support scientific experiments in a Moon-like outdoor environment, taking the components of our rover prototype one step further towards real space missions.

For future work, we aim to move further parts of our stereo image processing pipeline to FPGAs in order to unburden the LRU's main on-board computer as well as to allow for future space qualification of these components. In addition, we plan to extend the rover's object manipulation abilities by employing a more flexible grasp and assembly planning and by implementing whole-body planning and control. A future 6D whole-body local path planning could also allow the LRU to safely overcome tougher rough-terrain obstacles. Furthermore, we work on joint localization and mapping with multiple robots to improve efficiency through parallelization, robustness through redundancy and to benefit from complementary capabilities in heterogeneous robot teams. We intend to add additional semantic interpretations of the robots' surroundings, their own capabilities, and objects of interest to semantic maps and knowledge bases in order to support autonomous exploration, manipulation and task distribution between several robots.

Acknowledgements We thank the members of the Mobile Robots Group at DLR-RMC, especially Annika Maier, Bertram Willberg, Florian Schmidt and Philipp Lutz as well as our system administrators, in particular Stefan Engelhardt and Stefan von Dombrowski for their assistance. We thank Dr. Máximo A. Roa, Prof. Michael Beetz PhD, PD Dr. habil. Rudolph Triebel and Prof. Dr. Alin Albu-Schäffer for their support and many valuable discussions. We thank the anonymous reviewers for their insightful comments and suggestions.

Funding Information This work was supported by the Helmholtz Association, project alliance ROBEX (contract number HA-304) and partially funded by the DLR Space Administration.

Compliance with Ethical Standards

Conflict of interests The authors declare that they have no conflict of interest.

References

1. ROBEX - Robotic Exploration of Extreme Environments. http://www.robex-allianz.de/en/. Accessed 30 Aug 2016
2. DLR SpaceBot Camp. https://web.archive.org/web/20160320051557/ http://www.dlr.de/rd/en/desktopdefault.aspx/tabid-8101/13875_read-35268/ (2015). Accessed 31 Aug 2016
3. DLR SpaceBot Camp 2015 - Weltraumroboter live erleben. http://www.dlr.de/dlr/desktopdefault.aspx/tabid-10212/332_read-15747/#/gallery/21160 (2015). Accessed 30 Aug 2016
4. DARPA Robotics Challenge. http://www.darpa.mil/program/darpa-robotics-challenge (2016). Accessed 30 Aug 2016
5. Demo-Missions - ROBEX. http://www.robex-allianz.de/en/about-robex/demo-missions/ (2016). Accessed 24 Aug 2016
6. Google Lunar XPRIZE. http://lunar.xprize.org/ (2016). Accessed 30 Aug 2016
7. Lunar Reconnaissance Orbiter Camera. http://wms.lroc.asu.edu/lroc/rdr_product_select (2016). Accessed 24 Aug 2016
8. Manipulation - Kinova. http://www.kinovarobotics.com/assistive-robotics/products/manipulation/ (2016). Accessed 30 Aug 2016
9. Mission to the Moon. http://www.audi.com/corporate/en/innovations/mission-to-the-moon.html (2016). Accessed 30 Aug 2016
10. Mosh: the mobile shell. https://mosh.mit.edu/ (2016). Accessed 30 Aug 2016
11. Products-F&P. http://www.fp-robotics.com/en/technology/ (2016). Accessed 30 Aug 2016
12. ROBEX Field Trip - ROBEX. http://www.robex-allianz.de/en/field-trip/ (2016). Accessed 24 Aug 2016
13. Albu-Schäffer, A., Ott, C., Hirzinger, G.: A unified passivity-based control framework for position, torque and impedance control of flexible joint robots. Int. J. Robot. Res. **26**(1), 23–39 (2007)
14. Avsar, C., Frese, W., Meschede, T., Brieß, K.: Developing a planetary rover with students: Space education at TU Berlin. J. Autom. Mob. Robot. Intell. Syst., 8 (2014)
15. Beetz, M., Jain, D., Mösenlechner, L., Tenorth, M., Kunze, L., Blodow, N., Pangercic, D.: Cognition-enabled autonomous robot control for the realization of home chore task intelligence. Proc. IEEE **100**(8), 2454–2471 (2012)
16. Brand, C., Schuster, M.J., Hirschmüller, H., Suppa, M.: Stereo-vision based obstacle mapping for indoor/outdoor SLAM. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1846–1853. IEEE, Chicago (2014). https://doi.org/10.1109/IROS.2014.6942805
17. Brand, C., Schuster, M.J., Hirschmüller, H., Suppa, M.: Submap matching for stereo-vision based indoor/outdoor SLAM. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 5093-5100. IEEE, Hamburg (2015). https://doi.org/10.1109/IROS.2014.6942805
18. Brooks, R.A.: Intelligence without representation. Artif. Intell. **47**(1-3), 139–159 (1991)
19. Brunner, S.G., Steinmetz, F., Belder, R., Doemel, A.: RAFCON: A graphical tool for engineering complex, robotic tasks. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Deajeon (2016)
20. Brunner, S.G., Steinmetz, F., Belder, R., Dömel, A.: RAFCON: A graphical tool for task programming and mission control. In: RoboCup 2016: Robot Soccer World Cup XX, Lecture Notes in Computer Science. Springer, Leipzig (2016)
21. Büttner, S., Márton, Z.C., Hertkorn, K.: Automatic scene parsing for generic object descriptions using shape primitives. Robot. Auton. Syst. **76**, 93–112 (2016). https://doi.org/10.1016/j.robot.2015.11.003
22. Byrne, C. The Moon's Near Side Megabasin and Far Side Bulge, 1st edn. Springer-Verlag, New York (2013)

23. Czeluschke, A., Knapmeyer, M., Sohl, F., Bamberg, M., Lange, C., Luther, R., Margonis, A., Rosta, R., Schmitz, N.: Robex Asn study team: The ROBEX-ASN - a concept study for an active seismic network on the moon. In: European Planetary Science Congress 2014 (EPSC), vol. 9, pp. EPSC2014–728. Cascais, (2014)
24. Diankov, R., Kuffner, J.: OpenRAVE: A Planning Architecture for Autonomous Robotics. Robotics Institute, Pittsburgh. Tech. Rep. CMU-RI-TR-08-34, Pittsburgh (2008)
25. Dietrich, A., Wimböck, T., Albu-Schäffer, A., Hirzinger, G.: Reactive whole-body control: Dynamic mobile manipulation using a large number of actuated degrees of freedom. IEEE Robot. Autom. Mag. **19**(2), 20–33 (2012)
26. Eich, M., Hartanto, R., Kasperski, S., Natarajan, S., Wollenberg, J.: Towards coordinated multirobot missions for lunar sample collection in an unknown environment. J. Field Robot. R. **31**(1). https://doi.org/10.1002/rob.21491 (2014)
27. Gonzalez-Banos, H.H., Latombe, J.C.: Navigation strategies for exploring indoor environments. Int. J. Robot. Res. **21**(10–11), 829–848 (2002)
28. Grotzinger, J.P., et al.: Mars science laboratory mission and science investigation. Space Sci. Rev. **170**(1), 5–56 (2012)
29. Haarmann, R., Hofmann, P., Richter, L., Claasen, F., Apfelbeck, M., Klinkner, S., Schwendner, J.: Mobile Payload Element (MPE): concept study for a sample fetching rover for the ESA Lunar Lander Mission. Planet. Space Sci. **74**(1), 283–295 (2012)
30. Hellerer, M., Bellmann, T., Schlegel, F.: The DLR visualization library - recent development and applications. In: Proceedings of the 10th International Modelica Conference, pp. 899–911. Linköping University Electronic Press; Linköpings universitet, Lund (2014). https://doi.org/10.3384/ecp14096899
31. Hellerer, M., Schuster, M.J., Lichtenheldt, R.: Software-in-the-loop simulation of a planetary rover. In: The International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS). Beijing (2016)
32. Heppner, G., Roennau, A., Oberländer, J., Klemm, S., Dillmann, R.: Laurope - six legged walking robot for planetary exploration participating in the spacebot cup. In: International Conference on Automation and Robotics in Space (ASTRA). Noordwijk (2015)
33. Hirschmüller, H.: Stereo processing by semiglobal matching and mutual information. IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI) **30**(2), 328–341 (2008)
34. Hirschmüller, H., Innocent, P.R., Garibaldi, J.M.: Fast, unconstrained camera motion estimation from stereo without tracking and robust statistics. In: International Conference on Control, Automation, Robotics and Vision (ICARCV). Singapore. https://doi.org/10.1109/ICARCV.2002.1238577 (2002)
35. Hirzinger, G., Sporer, N., Albu-Schaffer, A., Hahnle, M., Krenn, R., Pascucci, A., Schedl, M.: Dlr's torque-controlled light weight robot III-are we reaching the technological limits now? In: IEEE International conference on robotics and automation (ICRA), vol. 2, pp. 1710–1716. IEEE (2002)
36. Holz, D., Basilico, N., Amigoni, F., Behnke, S.: Evaluating the efficiency of frontier-based exploration strategies. In: International Symposium on Robotics (ISR) and the German Conference on Robotics (ROBOTIK). VDE, Munich (2010)
37. Holz, D., Behnke, S.: Registration of non-uniform density 3D point clouds using approximate surface reconstruction. In: International Symposium on Robotics (ISR) and the German Conference on Robotics (ROBOTIK), Munich (2014)
38. Hornung, A., Wurm, K.M., Bennewitz, M., Stachniss, C., Burgard, W.: OctoMap: An efficient probabilistic 3D mapping framework based on octrees. Auton. Robot. https://doi.org/10.1007/s10514-012-9321-0. http://octomap.github.com. Software available at http://octomap.github.com (2013)
39. Jennings, E., Seguí, J., Gao, J., Clare, L., Abraham, D.: The impact of traffic prioritization on deep space network mission traffic. In: IEEE Aerospace Conference (2011)
40. Kaess, M., Johannsson, H., Roberts, R., Ila, V., Leonard, J.J., Dellaert, F.: iSAM2 : Incremental smoothing and mapping using the Bayes tree. Int. J. Robot. Res. **31**, 217–236 (2012)
41. Klem, S.M., Henriksen, M.R., Stopar, J., Boyd, A., Robinson, M.S.: Controlled LROC narrow angle camera high resolution mosaics. In: Lunar Planetary Science Conference. The Woodlands (2014)
42. Kriegel, S., Rink, C., Bodenmüller, T., Suppa, M.: Efficient next-best-scan planning for autonomous 3D surface reconstruction of unknown objects. J. Real-Time Image Process. **10**(4), 611–631 (2015). https://doi.org/10.1007/s11554-013-0386-6
43. Kubota, T., Otsuki, M., Shimada, T., Kuroda, Y., Kunii, Y.: Test-beds rovers for planetary surface exploration. In: International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS). Turin (2012)
44. LaValle, S.M., Kuffner, J.J. .Jr.: Rapidly-exploring random trees: Progress and Prospects. Algorithmic and Computational Robotics: New Directions pp. 293—308 (2000)
45. Lichtenheldt, R., Hellerer, M., Barthelmes, S., Buse, F.: Heterogeneous , multi-tier wheel ground contact simulation for planetary exploration. In: ECCOMAS Thematic Conference on Multibody Dynamics. Barcelona (2015)
46. Maimone, M., Johnson, A., Cheng, Y., Willson, R., Matthies, L.E.M.H., Khatib, O.: Experimental Robotics IX, chap. Autonomous Navigation Results from the Mars Exploration Rover (MER) Mission, pp. 3–13. Springer, Berlin (2006)
47. Ott, C.: Cartesian Impedance Control of Redundant and Flexible-Joint Robots, Spr. Tra. Adv. Robot, vol. 49. Springer, Berlin (2008)
48. Porges, O., Stouraitis, T., Borst, C., Roa, M.A.: Reachability and capability analysis for manipulation tasks. In: ROBOT2013: First Iberian Robotics Conference, pp. 703–718. Springer, Madrid (2014)
49. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y.: ROS: An open-source robot operating system. In: Workshop on Open Source Software (ICRA), vol. 3, p. 5. Kobe, Japan (2009)
50. Reill, J., Sedlmayr, H.J., Kuß, S., Neugebauer, P., Maier, M., Gibbesch, A., Schäfer, B., Albu-Schäffer, A.: Development of a mobility drive unit for low gravity planetary body exploration. In: Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA). Noordwijk (2013)
51. Reill, J., Sedlmayr, H.J., Neugebauer, P., Maier, M., Krämer, E., Lichtenheldt, R.: MASCOT - asteroid lander with innovative mobility mechanism. In: Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA). Noordwijk (2015)
52. Rusu, R.B., Cousins, S.: 3D is here: Point Cloud Library (PCL). In: IEEE International Conference on Robotics and Automation (ICRA). Shanghai (2011)
53. Schadler, M., Stückler, J., Behnke, S.: Data set Spacebot Arena. http://www.ais.uni-bonn.de/mav_registration/. Accessed 30 Aug 2016 (2014)
54. Schaub, A., Hellerer, M., Bodenmüller, T.: Simulation of artificial intelligence agents using Modelica and the DLR visualization library. In: International Modelica Conference. Fürstenfeldbruck (2012)
55. Schmid, K., Ruess, F., Burschka, D.: Local reference filter for life-long vision aided inertial navigation. In: International Conference on Information Fusion (FUSION). IEEE, Madrid (2014)
56. Schmid, K., Ruess, F., Suppa, M., Burschka, D.: State estimation for highly dynamic flying systems using key frame Odometry with

varying time delays. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Vilamoura (2012)

57. Schneider, F.E., Wildermuth, D., Wolf, H.L.: ELROB and EURATHLON: Improving search & rescue robotics through real-world robot competitions. In: International Workshop on Robot Motion and Control (RoMoCo), pp. 118–123. IEEE, Poznan (2015)
58. Schuster, M.J., Brand, C., Brunner, S.G., Lehner, P., Reill, J., Riedel, S., Bodenmüller, T., Bussmann, K., Büttner, S., Dömel, A., Friedl, W., Grixa, I., Hellerer, M., Hirschmüller, H., Kassecker, M., Márton, Z.C., Nissler, C., Ruess, F., Suppa, M., Wedler, A.: The LRU rover for autonomous planetary exploration and its success in the SpaceBotCamp challenge. In: IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC). Bragança (2016)
59. Schuster, M.J., Brand, C., Hirschmüller, H., Suppa, M., Beetz, M.: Multi-Robot 6D graph SLAM connecting decoupled local reference filters. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Hamburg (2015)
60. Schwendner, J., Röhr, T., Haase, S., Wirkus, M., Manz, M., Arnold, S., Machowinski, J.: The artemis rover as an example for model based engineering in space robotics. In: Workshop Proceedings of the IEEE International Conference on Robotics and Automation (ICRA). Hong Kong (2014)
61. Senarathne, P.G.C.N., Wang, D.: Frontier based exploration with task cancellation. In: IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), pp. 1–6. IEEE, Toyakocho (2014)
62. Sentis, L., Khatib, O.: Synthesis of whole-body behaviors through hierarchical control of behavioral primitives. Int. J. Hum. Robot. **2**(4), 505–518 (2005)
63. Shreiner, D.: OpenGL programming guide: The official guide to learning openGL, Versions 3.0 and 3.1, 7th edn. Addison-Wesley Professional (2009)
64. Strobl, K.H., Hirzinger, G.: Optimal hand-eye calibration. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Beijing (2006)
65. Stückler, J., Schwarz, M., Schadler, M., Topalidou-Kyniazopoulou, A., Behnke, S.: NimbRo Explorer: Semi-autonomous exploration and mobile manipulation in rough terrain. J. Field Robot. (2015)
66. Sünderhauf, N., Neubert, P., Truschzinski, M., Wunschel, D., Pöschmann, J., Lange, S., Protzel, P.: Phobos and Deimos on mars - two autonomous robots for the DLR SpaceBot cup. In: Proceedings of International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS). Montreal (2014)
67. Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G.: Stanley: The robot that won the DARPA grand challenge. J. Field Robot. **23**(9), 661–692 (2006)
68. Tran, T., Rosiek, M.R., Beyer, R.A., Mattson, S., Howington-Kraus, E., Robinson, M.S., Archinal, B.A., Edmundson, K., Harbour, D., Anderson, E.: Generating digital terrain models using LROC NAC images. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences (ISPRS Archives), **38** (2010)
69. Washington, R., Golden, K., Bresina, J., Smith, D., Anderson, C., Smith, T.: Autonomous rovers for mars exploration. In: IEEE Aerospace Conference, vol. 1. Snowmass at Aspen (1999)
70. Wedler, A., Chalon, M., Landzettel, K., Görner, M., Krämer, E., Gruber, R., Beyer, A., Sedlmayr, H.J., Willberg, B., Wieland, B., Reill, J., Grebenstein, M., Schedl, M., Albu-Schäffer, A., Hirzinger, G.: DLR's dynamic actuator modules for robotic space applications. In: Aerospace Mechanisms Symposium. Passadena - Hilton -JPL (2012)
71. Wedler, A., Maier, A., Reill, J., Brand, C., Hirschmüller, H., Schuster, M.J., Suppa, M., Beyer, A., Lii, N.Y., Maier, M., Sedlmayr, H.J., Haarmann, R.: Pan/Tilt-Unit as a perception module for extra-terrestrial vehicle and landing systems. In: Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA). Noordwijk (2013)
72. Wedler, A., Rebele, B., Reill, J., Suppa, M., Hirschmüller, H., Brand, C., Schuster, M., Vodermayer, B., Gmeiner, H., Maier, A., Willberg, B., Bussmann, K., Wappler, F., Hellerer, M., Lichtenheldt, R.: LRU - lightweight rover unit. In: Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA). Noordwijk (2015)
73. Wettergreen, D., Wagner, M., Jonak, D., Baskaran, V., Deans, M., Heys, S., Pane, D., Smith, T., Teza, J., Thompson, D.R., Tompkins, P., Williams, C.: Long-distance autonomous survey and mapping in the robotic investigation of life in the atacama desert. In: Proceedings of International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS). Hollywood (2008)
74. Yamauchi, B.: Frontier-based exploration using multiple robots. In: Int. Conf. Autonom. Agents, pp. 47–53. ACM, Minuenpolis (1998)
75. Zacharias, F., Borst, C., Hirzinger, G.: Capturing robot workspace structure: Representing robot capabilities. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3229–3236, San Diego (2007). https://doi.org/10.1109/IROS.2007.4399105
76. Ziegler, J., Bender, P., Schreiber, M., Lategahn, H., Strauss, T., Stiller, C., Dang, T., Franke, U., Appenrodt, N., Keller, C.G., Kaus, E., Herrtwich, R.G., Rabe, C., Pfeiffer, D., Lindner, F., Stein, F., Erbs, F., Enzweiler, M., Knöppel, C., Hipp, J., Haueis, M., Trepte, M., Brenk, C., Tamke, A., Hanaat, M.G., Braun, M., Joos, A., Fritz, H., Mock, H., Hein, M., Zeeb, E.: Making bertha drive - an autonomous journey on a historic route. IEEE Intell. Transp. Syst. Mag. **6**(2), 8–20 (2014)

Martin J. Schuster is a researcher and Ph.D. student at the Department of Perception and Cognition of the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). Before he joined DLR in 2012, he received his Master's degree in Computer Science from the Georgia Institute of Technology in Atlanta, USA in 2010 and his Bachelor's as well as a second Master's degree in Computer Science with minor subject Mechanical Engineering from the Technische Universität München (TUM), Munich, Germany in 2008 and 2011. In addition, he worked for the KnowDive research group at the Università di Trento, Italy in 2010 and received a Bachelor's degree in Philosophy from the Ludwig-Maximilians-Universität München (LMU), Munich, Germany in 2014. His current research interest lies on methods for localization and mapping with heterogeneous multi-robot teams in GPS-denied environments in the context of planetary exploration and search and rescue scenarios.

Sebastian G. Brunner is a researcher at the Department of Cognitive Robotics of the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). He received his Master's degree in Computer Science with major subject Artificial Intelligence and Robotics from the Technische Universität München (TUM), Munich, Germany in 2014. His current research interest lies in the field of mobile manipulation with the focus on task programming, task planning and scheduling.

Kristin Bussmann received her M.Sc. degree in mechanical engineering with focus on control theory and information theory from the Technische Universität München (TUM) in 2014. In the same year, she joined the Department of Analysis and Control of Advanced Robotic Systems at the Institute of Robotics and Mechatronics, German Aerospace Center (DLR) as a research assistant. Her current research interests lie in the field of mobile manipulation, particularly including compliance control methods for manipulators as well as the control of nonholonomic systems.

Stefan Büttner is doing his Master's degree in Computer Science at the University of Augsburg. Since 2014, he is working as a student at the Department of Perception and Cognition at the Institute of Robotics and Mechatronics, German Aerospace Center (DLR), his primary research topic being pose estimation of unknown objects. He received his B.Sc. degree in Mathematics from the University of Applied Science, Stuttgart in 2015. His fields of interest are computer vision, computer graphics, and virtual reality applications.

Andreas Dömel is a researcher at the Department of Autonomy and Teleoperation at the Robotics and Mechatronics Center, German Aerospace Center (DLR). He received his Dipl.-Ing. degree in electrical and information engineering in 2010 from the Technical University of Munich. Since 2010 he is with the Institute of Robotics and Mechatronics. His research interests are probabilistic motion planning and design of complex autonomous robotic systems.

Matthias Hellerer is a researcher and Ph.D. student at the Department of Space System Dynamics of the Institute of System Dynamics and Control, German Aerospace Center (DLR). Before he joined DLR in 2012, he received his Master's degree in Robotics, Cognition and Artificial Intelligence from the Technische Universität München (TUM), Munich, Germany in 2012. His current research interest lies on methods for the simulation and design of planetary rovers with a focus on the locomotion system and the wheel ground interaction.

Hannah Lehner is a researcher at the Department of Perception and Cognition at the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). She received her master's degree in Geodesy and Geoinformation Science, with the major subjects Computer Vision and Geoinformation Science, from the Technical University of Berlin in 2014. Her current research activities at DLR include methods for autonomous planetary exploration and task directed exploration in order to efficiently find known objects in unknown or partially unknown environments.

Peter Lehner is a researcher at the Department of Autonomy and Teleoperation at the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). Before he joined DLR in 2014, he received his master's degree in Computer Engineering from the Technical University of Berlin in 2014. His current research activities include developing methods for autonomous motion generation for mobile manipulation systems to empower mobile robots to autonomously interact with their environment.

Oliver Porges is a researcher at the Department of Autonomy and Teleoperation at the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). He received a double degree from Luleå Univeristy of Technology and Würzburg University in Space Science and Technology. His research interests are in robot workspace analysis (fixed and free-floating platforms), reactive motion planning with vision in the loop, autonomous grasping and vision for locomotion for bipedal and quadrupedal robots.

Josef Reill received the degree in electrical engineering from the Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany in 2005 and the Dr.-Ing. degree in 2010. From 2005 to 2009 he has been working with the chair of electrical drives and machines in Erlangen. His research was focused on the optimisation of vector and sensorless control of permanent magnet synchronous motors. Since November 2009 he is with the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). His main area of activity is the control of permanent synchronous machines used in robotic joints for terrestrial and space applications. He is member of the MOREX and ROBEX project team and project manager of MASCOT mobility subsystem.

Sebastian Riedel received his M.Sc in Computer Science from the Technical University Munich in early 2015. Since mid 2015 he works as research assistant at the Department of Cognitive Robotics at the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). Among his current research interests are experience- and data-driven approaches to fault detection and task execution monitoring as well as approaches for automatic and context-dependent algorithm parametrization.

Mallikarjuna Vayugundla is a research engineer at the Department of Perception and Cognition at the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). He received his Master's degree in Autonomous Systems from the Bonn—Rhein—Sieg University of Applied Sciences, Sankt Augustin, Germany in 2016. He is interested in autonomous mobile robot navigation especially for large and unstructured environments.

Bernhard Vodermayer is a senior research associate at the Department of Mechatronic Components and Systems of the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). He joined the institute in 2004, in 2006 he received his Diploma degree in Computer Science with major subject Electrical Engineering from the Technische Universität München (TUM), Munich, Germany. His current research interests are space based energy management, space based docking systems, polymer based actuators and sensors as well as robotic prostheses.

Tim Bodenmüller is senior researcher at the Department of Perception and Cognition of the Robotics and Mechatronics Center, German Aerospace Center (DLR). He received his Dipl.-Ing. degree in electrical and information engineering in 2001 from the Technical University of Darmstadt and his PhD in electrical engineering in 2009 from the Technical University of Munich. Since 2001, he is with the Institute of Robotics and Mechatronics, where he works in the field of 3D-sensing, -modeling, and -processing.

Christoph Brand received his B.S. Degree and Diploma in Electrical Engineering and Information Technology from the Technical University of Munich (TUM), Germany, in 2010 and 2011. He joined the Institute of Robotics and Mechatronics, German Aerospace Center (DLR) in 2011. His research interests include autonomous navigation, localization and mapping based on 3D sensor data considering unstructured and previously unknown environments.

Werner Friedl is the lead robotic hand engineer of the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). He received his Dipl.-Ing. degree in fine mechanics from the University of Applied Sciences, Munich, Germany, and joined the institute in 2004. His research interests include tendon driven robots and variable stiffness actuators.

Iris Grixa is a researcher at the Department of Perception and Cognition at the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). She received her diploma in Geography with the major subjects Remote Sensing and Hydrology, from the Ludwig-Maximilians University of Munich in 2009. Her current research activities and interests include methods for autonomous and biologically inspired navigation of mobile robots, especially for long-term operation.

Heiko Hirschmüller is co-founder and software engineer at Roboception GmbH in Munich, Germany. From 2003 until 2015 he worked as computer vision researcher at the Department of Perception and Cognition of the Institute of Robotics and Mechatronics, German Aerospace Center (DLR), where he also coordinated the mobile robotics group from 2009 until 2015 and served as leader of the team RMExplores! for the SpaceBotCamp in 2015. He received his Ph.D. degree on real time stereo vision from De Montfort University in Leicester, UK in 2003, his M.Sc. degree on Human Computer Systems from De Montfort University in Leicester, UK in 1997 and his Dipl.Inform. (FH) degree from Fachhochschule Mannheim, Germany in 1996. From 1997 until 2000 he worked as software engineer at Siemens AG Mannheim, Germany. Heiko Hirschmüller received the Carl-Pulfrich Award in 2011 and the DLR Wissenschaftspreis in 2009, both related to his development of the Semi-Global Matching method. He has also won several best paper awards as co-author. His main interests are in stereo vision, visual odometry and 3D reconstruction from images.

Michael Kaßecker is research assistant with the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). He studied computer science with minor psychology and graduated from Braunschweig University of Technology. After graduation he affiliated with the Technische Universität München and worked on natural language processing and cognition for robotics. In 2010 he concentrated on computer vision, object detection and cognition with the DLR and was member of several project teams.

Zoltán-Csaba Márton is a group leader for semantic scene analysis in the Perception and Cognition Department at the Institute of Robotics and Mechatronics, German Aerospace Center (DLR) since 2012. He received his PhD degree (summa cum laude) in 2012 from the Technische Universität München (TUM), where he worked as a researcher since 2007. He was one of the 3 original developers of the Point Cloud Library (PCL) at TUM. He is a graduate of the Technical University of Cluj Napoca, where he obtained a Dipl. Ing. (M.Sc equivalent) degree in control engineering.

Christian Nissler is a researcher at the Department of Perception and Cognition at the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). Before he joined DLR in 2011, he received his Diploma in Technical Cybernetics from the University of Stuttgart in 2011. His current research activities at DLR include methods for precise and robust pose estimation in both structured and unstructured environments.

Felix Ruess is a software engineer at Roboception GmbH in Munich. From 2010 until 2016 he worked as a researcher at the Department of Perception and Cognition of the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). He received his Masters degree in Robotics, Cognition and Artificial Intelligence from the Technische Universität München (TUM), Germany in 2012. His main interests lie in system design, robotic frameworks, probabilistic data fusion and navigation for embedded systems.

Michael Suppa received his master’s degree (Dipl.-Ing.) in Electrical Engineering from the University of Hannover with focus on Automation/Mechatronics with magna cum laude in 2000. In 2007, he received his Doctoral Degree (Dr.-Ing.) in Mechanical Engineering from the University of Hannover with summa cum laude. The title of his thesis is “Autonomous Robot Work Cell Exploration using Multisensory Eye-in-Hand Systems”. From 2000 until 2003, Dr. Suppa gained experience as project manager and researcher at the Institute of Robotics and Mechatronics, German Aerospace Center (DLR) in the research areas: robotic exploration, 3-D vision, and data fusion. During this time he was visiting researcher at Simon Fraser University, Robotic Algorithms and Motion Planning Lab (RAMP), Burnaby, Canada. From 2004 until 2009, he was coordinator of the 3-D modelling and vision group at the institute. From 2009 until 2015, he was Head of Department Perception and Cognition, managing over 40 employees. The group is a recognized world leader on key robotic research topics such as complex scene analysis, perception for resource-limited systems, robotic cognition. Since 2013, he coordinates the topic group perception with SPARC and was appointed deputy institute director of the Institute of Roboctics and Mechatronics in August 2015. Dr. Suppa was PI in the EC project TAPAS and has taken part in several other European and national projects. Dr. Suppa published over 60 papers and was nominated for and received several best paper awards. In March 2015 he co-founded Roboception, a DLR spin-off company devoted to advance the state of the art in 3D sensors and vision. Besides being CEO of Roboception he is also responsible for business models as well as strategy and technology development.

Armin Wedler received his Diploma in Mechanical Engineering and Bachelor in Robotics from Leibniz University of Hanover in 2004 and his PhD on “Adaptive passive compliant systems for use in robotics” in 2010. Starting in 2006, he worked for Leibniz University of Hanover until he joined the Institute of Robotics and Mechatronics, German Aerospace Center (DLR) in 2008. Since then, he has been focusing on the design and development of advanced space robotics, planetary exploration and mobile systems. He has worked as project manager, technical manager and member of project teams for numerous industrial and scientific robotic projects including ExoMars, DEXHAND, MPE, NLL, TeaM, MOREX and ROBEX where he has become the representative spokesperson of the project (since 2014). Furthermore Armin Wedler is active in the ISECG group (since 2014) and coordinator of the DLR-RMC internal groups for mobile robots (since 2015) and for the planetary exploration group (since 2014).