SIMULATION OF THE ROBOTIC LUNAR PROTOTYPE MIRA3D

Maximilian von Unwerth, Anna Voß, Simon Stapperfend, Stefan Linke, Enrico Stoll
Technische Universität Braunschweig, Institute for Space Systems
Hermann-Blenk-Straße 23, 38108 Braunschweig, Deutschland
E-Mail: {*m.von-unwerth, a.voss, s.stapperfend, stefan.linke, e.stoll*}*@tu-braunschweig.de*

The Moon becomes more and more interesting. For the coming years, several new missions are planned and the idea of a Moon Village is not just science fiction anymore. As Jan Wörner consistently emphasises, the Moon offers a lot of opportunities. For example, it could be our listening post to outer space or our gateway to deep space missions [1]. To make it possible to implement such a complex mission, robotic systems such as the lunar robotic prototype MIRA3D are indispensable. Since tests with robotic systems, especially tests on autonomous vehicles, require great effort and bring a high risk for the hardware, proper simulations of the system have to be investigated. This paper describes the challenges and opportunities a simulation of a system like MIRA3D offers and gives an insight into the procedure of developing the necessary components.

Keywords: Robotics, Simulation, Rover, ISRU, ROS, URDF

1. MOTIVATION

In the view of prospective missions, establishing permanent bases on Moon and Mars, new technologies for in-situ resource utilization (ISRU) become increasingly important. Autonomous manipulation with mobile robots must bring the solution for long-term tasks, which have to be executed with high precision and the absence of any possibility of human intervention. In section 2, MIRA3D, a robot which carries a powder based - fused deposition modelling (PF-FDM) printing head [2] for the processing of lunar regolith to create structures like tools or whole habitats for a Moon Village, is described. The use of these local resources is the most economic way to realize a project like this. The processes could take several months. So, with the aim of printing a whole village using this technology, a lot of parallel collaborating robots are needed and autonomy will replace teleoperations as the most important concept to control a spacecraft on far extraterrestrial bodies in the future. Considering missions like the first flight of Ariane 5, or the Mars Climate Orbiter, especially systems for space applications have to be tested and validated in as many simulations as possible. This is crucial to prevent remediable mistakes causing system-failures and multi-billion-euro losses. While the first disastrous flight of Europeans new rocket 1996, an unhandled integer overflow in the inertial measurement unit led to the failure of the inertial navigation system and its backup-system, which caused the self-destruction triggered shortly thereafter [3]. Also a software bug in the Mars Climate Orbiter mission 1999 led to a disaester. The on-board computer of the spacecraft worked with English units and ground control with the metric system, which led to an essentially to low orbit at Mars [4]. As with Ariane 5, tests on the whole system are often cancelled for money and time issues or are too risky, too slow or simply not possible. For a similar fault-critical system like that of an autonomous rover, establishing a base on a far planets surface, reliable software is as essential as fail-safe hardware. A simulation also offers other advantages. Software and hardware can be developed in parallel and in a flexible way. Both can adopt to changes very fast and it is possible to rapidly prototype and test first subcomponents, like 3D-printhead configurations or algorithms for autonomous navigation, before manufacturing and final implementation. Additionally, more than one person can work with the robot, because the simulation software can be copied as often as necessary. A simulation of MIRA3D delivers the capability to demonstrate the entire process of building lunar habitats in a vivid view. The simulation model presented in this paper is furthermore designed to study the collaboration of several robots. In this context, a material collector and a processor robot could work together. Fur such tasks with multi-robot collaboration, the simulation described by this paper, may be used. In section 3 the development of the simulation for MIRA3D is described. MIRA3D uses the Robot Operating System (ROS) for control and thus, ROS is used in the simulation. This guarantees the interchangeability of the software and hardware without effecting changes in the whole systems behaviour. The whole robot is finally described by description files in the Unified Robot Description Format (URDF). To control the robot in the simulation, a user interface is provided by this work. Section 4 gives an overview about this interface. In section 5, an evaluation on the most influencing aspects, comparing the simulation model with the real world, is dedicated. Possible deviations and their consequences for the simulation are stated. Eventually section 6 gives a short overview about the treated topics and outlines visions for the future and what could be an extension to better understand the challenges of an autonomous rover for extraterrestrial manufacturing.

doi:10.25967/490161

2. MOBILE ROBOTIC ARM FOR 3D-PRINTING MIRA3D

MIRA3D (**M**ob**i**le **R**obotic **A**rm for **3D**-printing) is a robotic lunar prototype, depicted in figure 1. It is introduced in the project 3D4Space. The mission of 3D4Space is the development and analysis of technologies for additive layer manufacturing of lunar regolith by a mobile robotic platform. To achieve this, various objectives must be pursued: The robotic component and its interaction with the lunar regolith has to be investigated. Therefore, a testbed and discrete element method (DEM) simulations were prepared. For detailed analysis of a design for a printhead, which uses lunar regolith, the regolith simulants TUBS-M and TUBS-T are developed by the Institute of Space Systems (IRAS, Technische Universität Braunschweig) and studied in melting tests to find a suitable process [5]. To test and validate those detailed scientific results, the robotic lunar prototype is built up. 3D4Space started in 2017 under the direction of IRAS[1] at TU Braunschweig. It is an alliance of five institutes of three universities in Germany.

Figure 1: MIRA3D with a kinematic printhead dummy to test the robots behaviour with end-effector

MIRA3D consists of an Universal Robots UR-10[2] mounted to an Innok Heros built by Innok Robotics[3]. All computer components are stored in the control box mounted behind the robot arm. To perceive its environment, the robot is equipped with a Stereolabs ZED camera in the front and four black-and-white cameras attached to the robotic arm and the control box. In figure 1, a printhead dummy is attached to the robotic arm to simulate and test the kinematic behaviour of the real printhead. The robot is controlled with an external remote control, which allows the manual work with the hardware. It also provides the capability to switch from remote-control mode to computer-mode, where all commands are given by a connected computer via wireless communication. To acquire the features needed to try software on the real hardware, the robot is equipped with ROS. To enable the interchangeability of software, written for the simulated and the real robot, an important requirement for the simulation is to also use ROS for running algorithms on the robot.

3. SIMULATION OF ROBOTIC SYSTEMS

The preparation of a simulation for a robotic system consists of three modules. Firstly to choose the simulation environment, the software, which provides a dynamic simulation of the robot in an environment, where typical operation scenarios can be created. It also provides tools to control the robot and simulate sensors to run algorithms with artificial data. Secondly, the robot model itself. To provide realistic behaviour, an exact virtual model of the real robot has to be implemented. The accuracy of sizes and masses defines for what the simulation can be used for. In the first place the simulation of MIRA3D will be a kinematic simulation of the robot, because inertia matrices are not given from the robots manufacturer and have to be approximated by simple geometric shapes, like cylinders for wheels and robot arm components with an uniform distribution of mass. With this in mind, the general behaviour and the functionality of algorithms could be tested. Thirdly, a proper test environment for the virtual robot has to be created. Therefore typical usage sites have to be integrated in the simulation.

ROS = Plumbing + Tools + Capabilities + Ecosystem [6] is the so called ROS equation, shortly describing the Robot Operating System. ROS is an open-source meta operating system. It offers hardware drivers like an operating system and provides tools and libraries for the control of robots. There are a lot of already existing solutions for problems like path-planning and obstacle avoidance, which are already implemented and can be used with any robot working with ROS (Capabilities). Multiple languages like C++, Python and Java can be used to write own code for a robot. Different ROS programs, so called ROS nodes communicate via messages, sent (published) and received (subscribed) to communication channels, the ROS topics (Plumbing). This decentralized approach gives the opportunity to work with a lot of different

---

[1] Link to official 3D4Space website: `http://www.helipod.de/3d4space/index.php/de/`

[2] Universal Robots e-Series: `https://www.universal-robots.com/de/produkte/ur10-roboter/`

[3] Innok Robotics, Mobile Innovationen: `https://www.innok-robotics.de/`

systems, all running ROS nodes, which are exchanging messages. One node works as the ROS master, which manages the communication [7]. ROS software is organized in packages, in which the robot description or simulation configurations can be found. In addition ROS implies several tools, for example for the visualization and evaluation of data (Tools). An active community and the Open Source Robotics Foundation (OSRF) take care of the maintenance and improve and extend the framework (Ecosystem). In the period of July 2017 to July 2018 the scientific paper, introducing ROS ([6]), was cited more than 4500 times, an increase of 29 percent in comparison to the prior year [8] and ROS is used on more and more robots, which shows the importance of the framework in the robotic context. For example, NASAs Robonaut 2, a humanoid robot, in use between 2011 and 2018 on board of the International Space Station (ISS) used ROS since a software update in 2014 [9]. For this reasons, MIRA3D is also equipped with ROS and all components of the real hardware can be controlled using the framework [10].

3.1 Choice of a suitable simulation environment

To find suitable software to simulate the robot, a comparison of common simulators has been carried out. Literature research led to three simulation environments, which are common in the use of mobile robots [11]. These three are:

1. *Gazebo* is an open-source 3D-simulation environment, developed for robots in special conditions by the University of Southern California. It is fully integrated in ROS and delivers a bunch of already implemented robot models, like the UR-10, sensors and also complex objects and environments [12]. Robot models have to be described in URDF (see 3.2). Because Gazebo is a part of the ROS framework, the community is very large and it is easily possible to get support.[4]

2. *V-Rep* was developed by Coppelia Robotics[5] in 2014. It adverties through many different ways to program the simulated robot. For example, robots can be controlled using ROS nodes or programs, that use the RemoteAPI. V-Rep also provides a library with robots like the UR-10 and offers the option to use these integrated robots with external components. V-Rep offers user-friendly features for modelling of robots and environments, which is great for prototyping and setups of simulations [13].

3. *Webots*: is developed by Cyberbotics[6] and is especially known for its use at the Robocup and the robot Nao [14]. Webots is one of the most developed simulators. [15]. It provides realistic rendering of the robots and environments and works with several programming languages. The software has a free trial version, but has to be bought for further usage.

To compare these three simulation environments different sources are used. [16] gives a very detailed overview about advantages and disadvantages of Gazebo and V-Rep. The authors claim, V-Rep offers the largest repertoire of features. However they evaluate V-Rep as the most resource-hungry simulator. Gazebo is open-source and it is strongly connected to the ROS framework, in comparison to the commercial simulator Webots. [17]. In summary Gazebo has been chosen to simulate MIRA3D, because it is the best suited way for a simulation of a mobile robot, running ROS and working open-source.

3.2 The virtual robot model

A simulation of a robot needs a model of the robot itself. To get a virtual model of a robot, software and hardware components have to be abstracted. Both have to be as realistic as possible to provide the interchangeability of the results for the real and the virtual robot. For example, an algorithm running on the simulated robot should be possible to run on the real robot, too. If this algorithm also offers the same result as on the simulated robot, the simulation could be very helpful to solve the real world robot problems with the virtual robot.

Robotic structures consist of links and joints. Links are all rigid bodies of the mechanism and joints the interconnections between links. The Unified Robot Description Format (URDF) [18] allows the description of all tree-like robot structures in a XML-like format, which is human-readable and very easy to process. URDF offers, next to the physical description of the robot, an interface to ROS for using sensors and actuators of the robot. Thus, it is the common format to model robots, when working with ROS [19]. MIRA3D consists of almost forty links and joints. Links can consist of simple shapes, like cubes or spheres or can be figured out by Standard Transformation Language (STL) or Digital Asset Exchange (DAE) files. While the models of the Innok Heros and UR-10 are provided by the manufacturers, the mounting rails, the control box and the components like the emergency-off button were remodelled with different construction

---

[4] Gazebo - Robot simulation made easy: `http://gazebosim.org/`

[5] V-Rep virtual robot experimentation platform: `http://www.coppeliarobotics.com/`

[6] Webots, Cyberbotics :`https://www.cyberbotics.com/`

[7] Blender - The free and open-source 3D creation suite: `https://www.blender.org/about/`

[8] FreeCAD - Your own 3D parametric modeler: `https://www.freecadweb.org/`

software like Blender[7] or FreeCAD[8]. All components are equipped with appropriate textures for demonstration purposes. Each link of a URDF robot description requires a visual and a collision geometry. To make the simulation faster and repeatedly runnable, the collision geometry is defined by a smaller STL-file, which defines the boundaries of the body for collision calculation. The visual component carries a DAE-file with a more detailed mesh geometry and textures, which can just be defined in such a DAE file. The use of STL files as collision bodies is always a compromise of accuracy and simulation performance. URDF defines the structure of a robot with parent-child relationships in each joint. The component-tree is built on the main part, in this case the Innok Heros main beam under the mounting plate, with the control box and the UR-10 on top of it. Both, visual and collision body needs an origin, the location where they are mounted on the parent part and how they are transformed relative to it. A joint defines a motion constraint between parent and child. In URDF also forces and velocity limits can be set. MIRA3D owns three different types of joints: A fixed joint, where parent and child are fixed and can not move relative to each other. Revolute joints with one degree of freedom, connecting two components via one rotation axis, which can be found in the UR-10 and continuous joints (revolute joints without any limits) for the wheels. Figure 2 describes the relation of visual and collision bodies and the relation of the parent and child component of a joint. The origins of each component were surveyed and the necessary joints were applied.

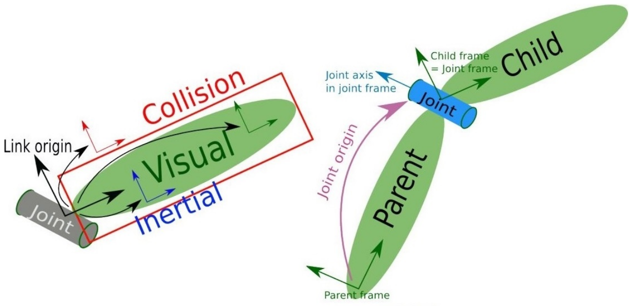

Figure 2: Origins of visual and collision element and the relation of parent and child link connected by a joint. [20]

In the links description the physical properties of the bodies are set. The mass can be obtained by weighing the hardware, but the inertia matrix has to be approximated by simple geometries, because the distribution of mass in the robots components is unknown. Similar to the collision body approximation through shapes, like cylinders for the wheels and cubes for the mounting plate or the beam link, the distribution of mass is approximated as equal over the whole body and the components were assumed to be basic shapes. The inertia matrix J (Equation 2) of the control box is for example calculated by Equation 1. The box has a mass m_{box} of 35 kilogram and a height $h_{box} = 300mm$. The box length is $l_{box} = 500mm$.

$$\begin{aligned} J_{XX} = J_{YY} &= \frac{1}{12} \cdot m_{box} \cdot ({h_{box}}^2 + {l_{box}}^2) \\ &= \frac{1}{12} \cdot 35.000kg \cdot ((0.3m)^2 + (0.5m)^2) \\ &= 0.992kgm^2 \\ J_{ZZ} &= \frac{1}{12} \cdot m_{box} \cdot ({l_{box}}^2 + {l_{box}}^2) \\ &= 1.458kgm^2 \end{aligned} \tag{1}$$

$$J = \begin{pmatrix} 0.992 & 0 & 0 \\ 0 & 0.992 & 0 \\ 0 & 0 & 1.458 \end{pmatrix} \tag{2}$$

More about the physical properties and MIRA3Ds simulation model can be found in [21]. Design and construction of the printhead prototype can be found in [22]. To make the robot description more modular and reduce redundancy, XML macro files (XACRO) were created. The XACRO description simplifies the URDF files. The differences of URDF and XACRO are explained in [20]. MIRA3Ds XACRO description was built in a modular way, so that new components can be integrated with less effort. The final version of the printhead will replace the currently used printhead demonstrator and therefore a new XACRO file can be created and integrated in the robots model. The old version could be commented out to fast undo for further tests. Also new sensors can be installed in the same way. This modular and expandable XACRO structure allows rapid prototyping, easy testing of different configurations and general maintenance of the simulation.

MIRA3D uses a variety of different sensors. Three black-and-white-cameras for navigation purposes are mounted at the backside of the control box. One points behind the rover, the other two are directed to MIRA3Ds wheels to control its environmental conditions. Another camera is attached to the robotic arm to enable an 360 ° view. A stereo-camera for obstacle detection and environment mapping is mounted to the frontal drive box and an inertial measurement unit is located in the control box to monitor orientation and improve odometry data via sensor fusion as in [23]. Details about choice of sensors and their arrangement are outlined in [24]. These sensors are also integrated in the XACRO robot description and send their data via ROS to existing subscribers. Thereby, the simulated robot uses the same topic as the real robot. The simulated camera parameters are set in accordance to the real sensors to enable the similarity of the processed data. The robots motors delivers odometry data, which can be used, but differ from reality very fast, because MIRA3D uses a differential drive, where all wheels can be controlled independently, which causes slip.

3.3 Environments and scenarios

To enable different test scenarios, diverse environments have been created. Here two examples will be described shortly. To make it possible to compare algorithms and the robots behaviour of reality and „virtuality“, the institutes floor with its stairs is created as a virtual environment. This also allows tests of the simulation in a simple environment without obstacles or slopes. This setting is used to try first path-planning and mapping algorithms. But the simulation offers more than a simple mapping of accessible locations. The simulation gives the chance to test the rovers properties and behaviour on the Moon. Therefore a so called *World* with a lunar setting was created. World files include terrain models, terrain texture, light sources, sky properties and properties like gravity and friction parameters. Gazebo can load terrain models as .dae-files or heightmaps. Heightmaps are image files where grey scales define the elevation of the simulated environment. The darker a pixel, the deeper the valley. The gazebo model catalogue offers a model of the Dune crater, three kilometres south of the Apollo 15 Landing Site [25], with an expansion of one by one kilometre. It also defines friction, gravity other physical settings. The models offers different slopes and the more than twenty metre deep crater Dune. The data for the heightmap comes from the Lunar Reconnaissance Orbiter (LRO), which analyses the Moon's surface since 2009 [26]. In figure 4, the black frame marks the contained area.

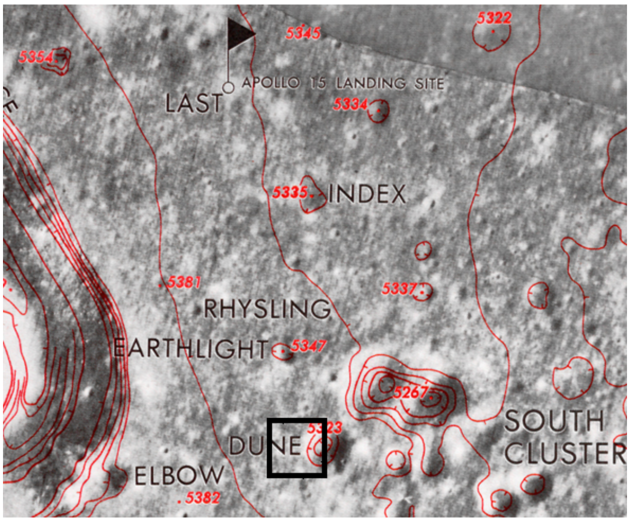

Figure 4: The region around the Dune crater, which is used in the simulation. [26]

Alternatively the National Aeronautics and Space Administration (NASA) offers 3D-models of the Apollo Landing Sites and different more detailed models of some craters and its surroundings [27], which can be used in the simulation by changing the models source in the .world file. Because the simulation should also be used for the purpose of demonstration, a starry sky with the Earth is built in. Because Gazebo is designed to simulated the Earth's sky, some parameters were changed to for example deactivate the Moon's or in this case the Earth's halo, which the Moon seems to have, because of the Earth's atmosphere. For further extension of the simulated scene, multiple rock structures were created. They can be easily inserted in the scene. The models were designed to resemble probes found at the dune crater, to make the simulation more realistic [28]. Figure 5 shows the final simulated model of MIRA3D with its printhead prototype in the lunar world simulated with Gazebo.

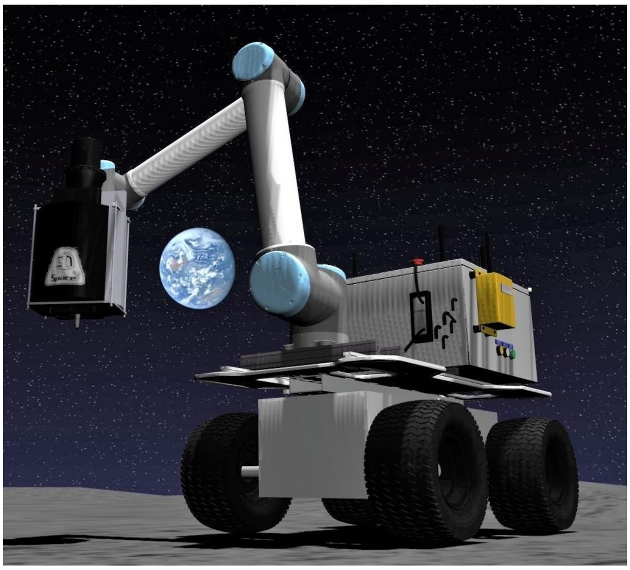

Figure 5: Simulated MIRA3D on the Moon to test the systems behaviour with less gravity.

4. GUIDED USER INTERFACES

A remote control is provided by the manufacturer. The commands given through the remote control are sent to the on-board computer and were processed as ROS messages. Thus, it was possible to integrate a mode to control the UR-10 with it, too. It is possible to control the whole robot just with the remote control. Additionally, the remote has an emergency switch which can stop the robot, even it is in autonomous computer controlled mode. The remote can also be used to control the robot in the simulation. To work with the robot without any hardware, a virtual „remote control“ was created with rqt, a QT-based framework for creation of user interfaces for ROS. The interface shows actual camera images and the values of the inertial measurement unit (IMU). With sliders, velocities for the wheels and targeted positions for all joints of the robot arm can be set. As well as the remote, this interface can be used to control the real hardware. Based on safety reasons, the remote control is always in contact with the robot and brings the capability to emergency shut-off the robot.

5. EVALUATION

The simulation of MIRA3D enables the work with the robot without using the real hardware. In this section a comparison and the applicability of the simulation for the development of software is discussed. For a geometric comparison of reality and the simulated robot, different postures of the robot are compared. The UR-10 is configured to reach different end-effector poses. With this procedure, some incorrect measurements and arrangements of antennas on the control box were found and fixed. To test the similarity of the robots behaviour, movement commands were sent using the interface described in section 4. Both robots executed their commands and reached similar goals in the floor environment. As indicated in section 3, odometry data does not reflect the correct distances and orientations. Complexer scenarios with slopes and obstacles led to unequal results. This is given by the approximated inertia properties for the components and the complexity of physics. But with this in mind, the simulation is very useful to implement algorithms for MIRA3D. Typical mapping and navigation tasks work with relative data, so that the robot reacts to environmental changes. This led to the same algorithmic behaviour which could be mapped from the simulation to the real world. According to tests, applied programs deliver same result when using the simulated robot and can be deployed to the real hardware with a little reconfiguration. First tests in the lunar environment shows, that small movements of the robotic arm has significant stronger effects on the whole system. The lower gravity led to further complex problems, which has to be investigated and solved in the view of using similar robotic systems in space. In summary, the simulation adequately abstracts the geometric properties and paves the way for analysis of the system MIRA3D and its operations. To simplify further work with MIRA3D, the whole process of creating this simulation is documented in [21].

6. CONCLUSION & OUTLOOK

In this paper, the process of implementing a simulation of the robotic lunar prototype MIRA3D is described. The paper summarizes the important parts of a robotic simulation. It gives an overview for how to choose the right software to fulfil the given requirements, like using the ROS framework. The paper outlines the modelling and aggregation of the virtual hardware and software components and the elements for a suitable environment. An evaluation on the most influencing aspect comparing reality and the simulation is conducted and algorithms for path-planning, mapping and collision avoidance are applied to both robots analysing similarities and differences. The simulation created for MIRA3D is designed to simplify the work with MIRA3D. In the future the integration of more test environments like quarry- or volcano-like areas could allow more test cases to develop sophisticated software for MIRA3D. As important as testing of the simulation is the maintenance. Software has to adopt to changes over time or it will become useless in the end [29]. For example, changed components of the real hardware have to be integrated in the simulation immediately. A changed arrangement of sensors will lead to totally different behaviour of algorithms. With the usage of ROS, software packages and utilized programs can be easily kept up-to-date on the hardware and the simulation. With this in mind, the simulation is very helpful to fulfil the goals of the project 3D4Space with the robot MIRA3D.

The in this paper presented simulation is furthermore developed to research the collaboration of several robots. In the context of creating a permanent base, at least a material collector and a processor robot have to interact, as depicted in figure 6. One robot will explore the immediate surrounding to mine lunar regolith and will bring it to the processor robot. This robot will undertake the task of the long-term printing process. In the future, a whole swarm of robots could undertake the great responsibility to autonomously transform a piece of the rough lunar landscape into a habitable area for humans. The necessary rover-rover interaction and the handling of multiple robots is studied by IRAS using this simulation. Further the printing process itself could be examined within the simulation, when the final version of the printhead gets finished. For such tasks, especially when more than one robot is needed or rapid prototypes are to be investigated, the simulation has to replace the expensive, time-consuming work with real hardware. It will help to develop software and hardware simultaneously and perform tests on the system in a low-cost, fast and safe way. Additionally, the simulation will help to study the challenges and opportunities such a system offers and will serve a knowledge transfer in the areas of autonomous robotic systems, the processing and recycling of materials in space conditions, as well as many other fields of science the project 3D4Space is assigned to. This work leads to further interesting topics in current research, like robot swarm collaboration, multiple rover supervision and the general challenges the autonomous creation of the first human outpost in space imply.

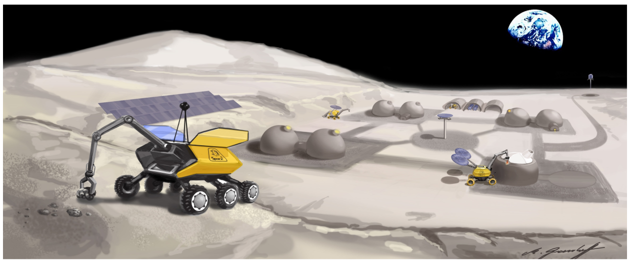

Figure 6: Artistic representation of the robots collecting and processing lunar regolith for habitats of a moon village. (Michael Grasshoff, ITD)

ACKNOWLEDGEMENTS

This work was supported by Prof. J. Steil of the Institute for Robotics and Process Control at TU Braunschweig.

References

[1] *Jan Wörner, Generaldirektor der ESA, über die weitere Erkundung des Weltraums, seine Vision eines Moon Village und 3D-Drucker in der Raumfahrt.* `https://m.esa.int/ger/ESA_in_your_country/Germany/Jan_Woerner_Moon_Village_-_Menschen_und_Roboter_gemeinsam_auf_dem_Mond`. – Retrieved 06.07.2019

[2] VOSS, A. ; KUHR, L. ; FREUND, R. ; LINKE, S. ; STOLL, E.: Requirements for a mobile lunar prototype for additive layer manufacturing. (2018)

[3] LIONS, Jacques-Louis ; LUEBECK, Lennart ; FAUQUEMBERGUE, Jean-Luc ; KAHN, Gilles ; KUBBAT, Wolfgang ; LEVEDAG, Stefan ; MAZZINI, Leonardo ; MERLE, Didier ; O'HALLORAN, Colin: *Ariane 5 flight 501 failure report by the inquiry board*. 1996

[4] LEVESON, N.: Role of software in spacecraft accidents. In: *Journal of spacecraft and Rockets* 41 (2004), Nr. 4, S. 564–575

[5] VOSS, Anna: Additive Fertigungsmethoden für die innovative Raumfahrt - 3D4Space - A short introduction into the activities of the Institute of Space Systems (IRAS). (2018), Mai

[6] OPEN SOURCE ROBOTICS FOUNDATION: *About ROS*. `http://www.ros.org/about-ros/`. – Retrieved 16.08.2019

[7] QUIGLEY, Morgan ; CONLEY, Ken ; GERKEY, Brian ; FAUST, Josh ; FOOTE, Tully ; LEIBS, Jeremy ; WHEELER, Rob ; NG, Andrew Y.: ROS: an open-source Robot Operating System. In: *ICRA workshop on open source software* Bd. 3 Kobe, Japan, 2009, S. 5

[8] FOOTE, T.: ROS - Community Metrics Report, Comparisons to July 2017 Report. (2018), Juli

[9] *Kapitel* ROS in Space: A Case Study on Robonaut 2. In: KOUBAA, A. (Hrsg.): *Robot Operating System (ROS) - The Complete Reference (Volume 1)*. Springer International Publishing, 2016, S. 343–374

[10] KUHR, L.: *T 1728 B - Evaluation of terrestrial lunar rover prototype designs and cost analysis of in-situ resource utilization*, Institut für Raumfahrtsysteme, Technische Universität Braunschweig, Bachelor Thesis, 2018

[11] IVALDI, S. ; PADOIS, V. ; NORI, F.: Tools for dynamics simulation of robots: a survey based on user feedback. (2014). – arXiv preprint arXiv:1402.7050

[12] CRAIGHEAD, Jeff ; MURPHY, Robin ; BURKE, Jenny ; GOLDIEZ, Brian: A survey of commercial & open source unmanned vehicle simulators. In: *Proceedings 2007 IEEE International Conference on Robotics and Automation* IEEE, 2007, S. 852–857

[13] NOGUEIRA, L.: Comparative analysis between gazebo and v-rep robotic simulators. (2014), S. 5. – Seminario Interno de Cognicao Artificial-SICA 2014

[14] ROBOCUP GERMAN OPEN 2019: *Standard Platform League (SPL)*. `https://www.robocupgermanopen.de/de/major/spl`. – Retrieved 01.08.2019

[15] HARRIS, Adam ; CONRAD, James M.: Survey of popular robotics simulators, frameworks, and toolkits. In: *2011 Proceedings of IEEE Southeastcon* IEEE, 2011, S. 243–249

[16] PITONAKOVA, L. ; GIULIANI, M. ; PIPE, A. ; WINFIELD, A.: Feature and performance comparison of the V-REP, Gazebo and ARGoS robot simulators. In: *Annual Conference Towards Autonomous Robotic Systems* Springer, 2018, S. 357–368

[17] STARANOWICZ, Aaron ; MARIOTTINI, Gian L.: A survey and comparison of commercial and open-source robotic simulator software. In: *Proceedings of the 4th International Conference on PErvasive Technologies Related to Assistive Environments* ACM, 2011, S. 56

[18] SUCAN, I. ; KAY, J.: *Unified Robot Description Format (URDF)*. `http://wiki.ros.org/urdf`. – Retrieved 04.09.2019

[19] QUIGLEY, M. ; GERKEY, B. ; SMART, W.: *Programming Robots with ROS: a practical introduction to the Robot Operating System*. OReilly Media, Inc., 2015

[20] WALCK, G.: Introduction to Robot Modeling in ROS - Understanding URDF and XACRO. In: *Universität Bielefeld* (2015), November

[21] VON UNWERTH, M.: *T 1813 B - Simulation des robotischen Mondroverdemonstrators MIRA3D*, Institut für Raumfahrtsysteme, Technische Universität Braunschweig, Bachelor Thesis, August 2018

[22] FOX, L.: T 1809 B - Konstruktion einer 3D-Druckkopfattrappe für Tests mit dem Roboterarm UR10. In: *Bachelor Thesis, Institut für Raumfahrtsysteme, Technische Universität Braunschweig* (2018)

[23] HUAI, J ; ZHANG, Y ; YILMAZ, A: Real-time large scale 3D reconstruction by fusing Kinect and IMU data. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 2 (2015), S. 491

[24] HARMS, S.: T 1814 S - Selection and commissioning of a sensor system for teleoperation of the lunar rover prototype MIRA3D. In: *Bachelor Thesis, Institut für Raumfahrtsysteme, Technische Universität Braunschweig* (SS2018)

[25] OKADA, K.: *Apollo 15 Landing Site 1000x1000*. `https://bitbucket.org/osrf/gazebo_models/src/default/apollo15_landing_site_1000x1000/`. Version: April 2015. – Retrieved 23.06.2019

[26] ARIZONA STATE UNIVERSITY: *LROC NAC Digital Terrain Model - Apollo 15 Landing Site*. `http://wms.lroc.asu.edu/lroc/view_rdr/NAC_DTM_APOLLO15`. Version: August 2013. – Retrieved 24.07.2019

[27] ERICKSON, K. ; BECK, B.: *NASA 3D Ressources*. `https://nasa3d.arc.nasa.gov/models`. Version: September 2017. – Retrieved 19.08.2018

[28] MEYER, C.: 15485 and 15486 - Vitrophyric Pigeonite Basalt. In: *Lunar Sample Compendium* (2010)

[29] CANFORA, Gerardo ; CIMITILE, Aniello: Software maintenance. In: *Handbook of Software Engineering and Knowledge Engineering: Volume I: Fundamentals*. World Scientific, 2001, S. 91–120