LEARNING-BASED MOTION CONTROL OF A ROVER ON UNKNOWN GROUND

Niklas Baldauf[(1)], **Toni Lubiniecki**[(1)], **Alen Turnwald**[(1)], **Kristin Lakatos**[(2)], **Nickolaos Panagiotopoulos**[(3)]

[(1)]*e:fs Techhub GmbH, {Niklas.Baldauf, Toni.Lubiniecki, Alen.Turnwald}@efs-techhub.com*
[(2)]*German Aerospace Center (DLR), Institute for Robotics and Mechatronics , Kristin.Lakatos@dlr.de*
[(3)]*European Space Research & Technology Centre (ESTEC/ESA) , Nickolaos.Panagiotopoulos@esa.int*

ABSTRACT

This paper presents the current status and scientific approaches from the ESA project DeLeMIS. Within that, e:fs and DLR Robotics investigate state-of-the-art approaches of self-learning methods to enable improvements in planetary space missions. The main objective of DeLeMIS is the autonomous navigation of a rover on unknown terrain without human intervention. To this end, algorithms from the latest research in the Artificial Intelligence (AI) and control engineering community are used for environmental perception and behaviour control. On the one hand, the rover is supposed to learn the ground conditions of different areas and their boundaries, both through camera inputs and through the interaction of the wheels with the ground. On the other hand, the system is intended to learn the best behaviour and the most appropriate strategy for its motion and navigation across different ground surfaces, taking into account safety and robustness aspects.

1 INTRODUCTION

In the project DeLeMIS, AI-methods are demonstrated that enable a planetary rover to move over uncertain terrain autonomously, i.e. without human intervention. This shall be possible even though limited information is available beforehand about the properties of the terrain. Several challenges arise from this problem, especially with regard to the motion control. In general, the physical behaviour of the rover, i.e. the motion and interactions with the environment, is very complex and cannot be fully captured by equations. Furthermore, the environment in which the rover navigates may be unknown and change over time. Therefore, conventional control methods, which require precise models, are more likely to fail. To this end, learning methods enabling improvements of the system model seem very promising.
By learning or intelligence in this context, we understand that the rover is able to acquire more information about itself and it's environment, change the representation of the knowledge accordingly, adapt the search for proper actions and even re-evaluate situations and actions through the newly acquired knowledge. To obtain more information, the rover is equipped with appropriate sensors.
The field of learning-based optimal control has been very active over the last decade and aims at combining aspects of system identification and learning with the actual control task. There are numerous approaches in the literature that intend to increase performance and robustness properties of optimization-based control approaches, such as MPC, even in the presence of unknowns and uncertainties.

As mentioned earlier, knowledge about the kinetics and kinematics of the rover and its interactions with the environment is at least partially and approximately available and can be expressed through differential equations. Control theory, and in particular optimal control, provides methods for computing actions that optimise a given objective function for such systems. We will mainly focus on Model Predictive Control (MPC), as this approach is a very well-developed and powerful system that can deal with nonlinear systems and constraints. MPC provides a set of advanced control methods for computing control inputs to a system such that optimal behaviour given by a cost function is achieved. The basics of the general nonlinear approach of MPC and the mathematical concepts and tools that are important for understanding and applying MPC are explained in [4] and [5]. However, since there are also unknown components of the physical model and disturbances, we will also use stochastic and learning-based extensions for the modelling and control framework. An overview of current learning-based MPC methods are provided in [12] and [14].
The motion control of the rover, the corresponding concepts, learning-based approaches and first results from the project are presented in this paper. Furthermore, the test and validation platform, including a simulation environment, is described which is also used to generate data for the ML approaches in the future work.

2 SCENARIO AND PROBLEM FORMULATION

The considered scenario is based on DLR's LRU (Lightweight Rover Unit) [3] in DLR's test facilities as shown in figure 2 and 3. A detailed description of the rover, its kinematics and the software structure of the system can be found in [7].

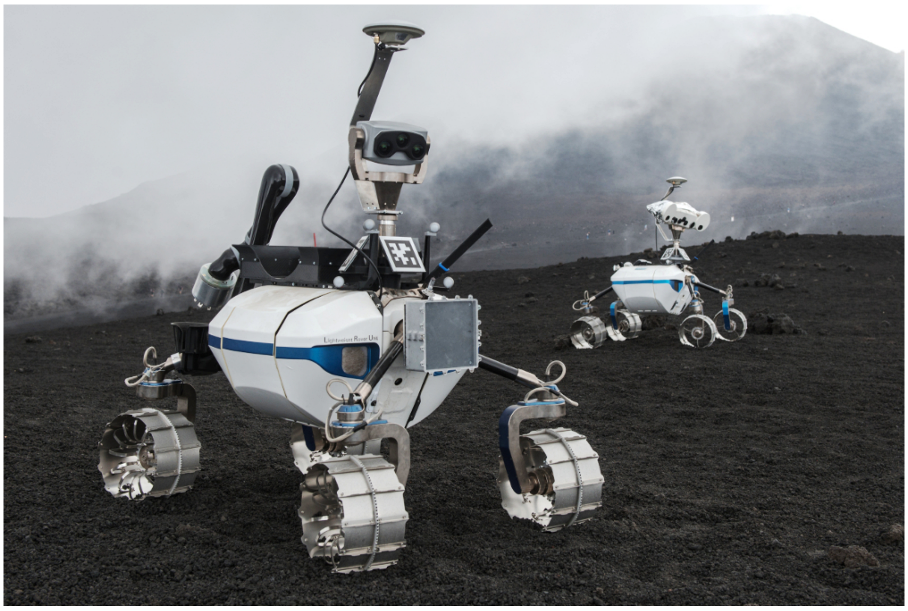

Figure 1: LRU1 and LRU2 on Mt. Etna, 2016. LRU2 features a robotic arm (front), LRU1 is equipped with additional scientific cameras (back).

The rover was designed as a small and lightweight research platform for planetary exploration scenarios, featuring a footprint of $114\text{cm} \times 74\text{cm}$. The focus of the platform design was to guarantee mobility in rough terrain. Therefore, the rover is equipped with four individually steerable wheels, which provide a velocity input in spinning direction. Additionally, the front wheels as well as the rear wheels are connected to the body via elastically suspended bogie axles, respectively. A pair of stereo cameras used for navigation purposes is mounted on the pan-tilt axis.

There exist two different variants of the LRU, see Fig. 1. LRU1 is equipped with a set of scientific cameras used among others for geological investigations during the ARCHES demo mission [13]. On the other hand, LRU2 has a 6-dof robotic manipulator arm mounted on the mobile platform in order to perform manipulation tasks.
A model of the LRU is also implemented in the simulation environment for rapid testing and data generation. The goal of the scenario definition is to provide a reproducible procedure to demonstrate and eventually evaluate whether a learning algorithm can improve the trajectory following behaviour of the rover.

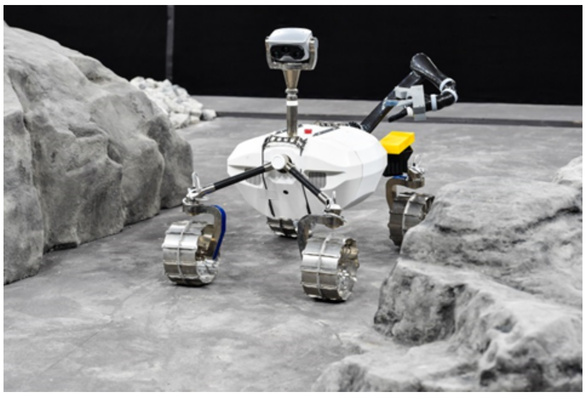
Figure 2: LRU operating in the DLR mobiliy lab.

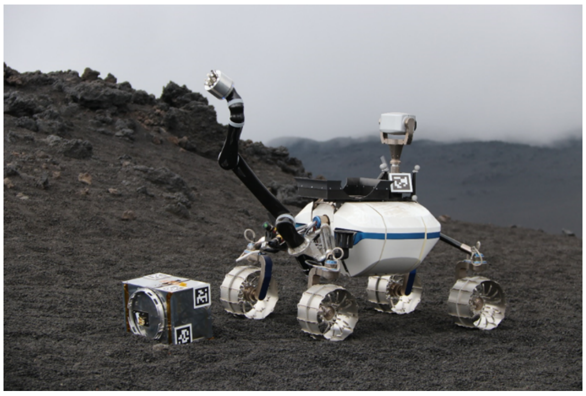
Figure 3: Rover operating in terrain on Mt. Etna.

The task of the motion control subsystem is to follow a given trajectory on a partially known or unknown ground. It is desired that the task is performed better after every iteration, which is taken as an indication for the learning. The considered scenario involves one of the testing facilities of DLR that the terrain features a significant profile of elevation change, impassable obstacles, such as large boulders or rocks, and at least two different kinds of soil, regarding their friction coefficient.

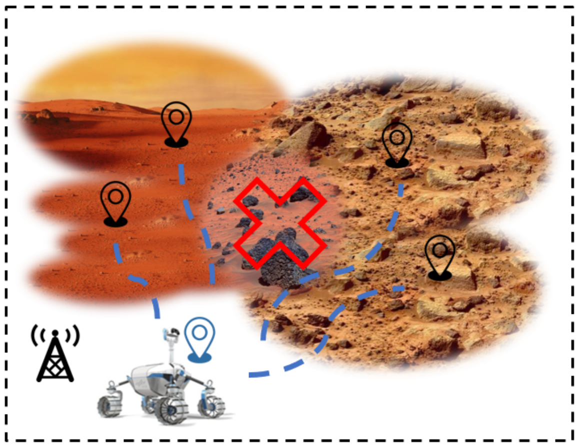
Figure 4: Exemplary structural concept of the scenario on the testing ground.

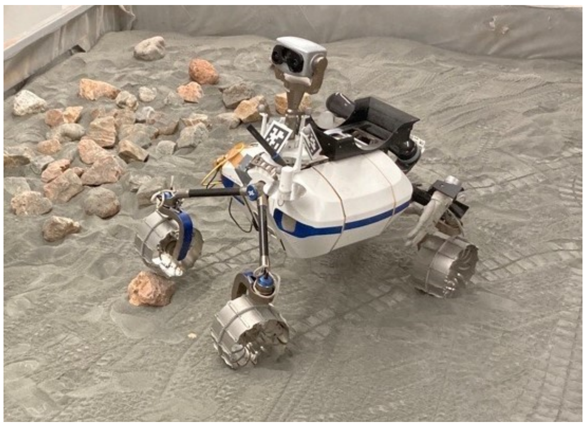
Figure 5: LRU in the planetary exploration lab at the DLR in Oberpfaffenhofen

As illustrated in Fig. 4, the rover is located at a defined starting position. In addition, there are several different target locations within the premises of the testing ground. All target locations are physically accessible for the rover by traveling along the a priori determined trajectories from the planning layer. The blue marker indicates the starting position of the rover, the black markers indicate different target locations. The red cross indicates impassable terrain. The blue dashed lines indicate the paths of the trajectories the rover should follow to reach the desired target location.

Regarding its ego localization, the rover utilizes the ground truth positioning system of the testing ground. This means, that the ego localization does not interfere with the evaluation of the improvements for each learning algorithm present below.
Before a scenario is started with one of the learning algorithms, the rover has the opportunity to carry out reference measurements by travelling along all the trajectories available so far in the terrain. In doing so, quantities such as position, speed, transverse and longitudinal distance to the trajectories, travel time as well as quantities representing the energy consumption and the robustness of the locomotion are recorded. By comparing these measurements with the evaluation measurements from the scenario, improvements in the behaviour of the trajectories tracking can be determined depending on improvement measures to be defined later. The implementation of each learning algorithm can be divided into three phases. However, each algorithm does not necessarily have to include all phases. In the first phase, learning is done offline from existing data. These data sets come, for example, from previous space missions, other experiments or data from simulation environments. The goal here is to create an initial learning state. In the second phase, the rover moves in close proximity to the lander to adapt or validate the learning algorithm to the current environment. This is similar to an installation run in a real space mission to check that all systems are working correctly and to validate parameters. The third phase is the continuous adaptation of the algorithm as it follows pre-planned trajectories to the target locations. In this context, it is about adjusting some parameters, units or functional structures within the algorithms using the collected data within a travel segment. In doing so, the initial learning state from the offline learning part and the land-based exploration part is constantly adjusted. In addition, ground information from the perception layer can also be taken into account if available. During this part, the rover follows all trajectories to visit each of the target locations, analogous to the procedure for performing reference measurements.
The evaluation is based on certain criteria that will be defined and detailed during the implementation phase of the project. Such criteria could be the accuracy of the trajectory tracking of the trajectory. Aspects such as the lateral deviation from the trajectory, the energy consumed for propulsion or the travel time can be taken into account. For each learning algorithm, the results of the executed scenario are compared with its previously performed reference measurements.
As mentioned before, the planning layer uses a priori calculated trajectories to the different destinations. This is necessary to ensure that the same sophisticated trajectory is used in the reference and scenario measurements. For example, a trajectory with lower altitude differences is likely to be less energy demanding.

3 SIMULATION ENVIRONMENT AND TEST PROCESS

For easier and faster development during the project, a simulation environment was created, including a model of the LRU rover, a lunar surface and a flexible interface architecture for the integration of the different algorithms. This Gazebo simulator (https://gazebosim.org/), which is embedded in the ROS framework (https://www.ros.org/), also serves as a data generator for the ML-based methods and as a test and verification platform for algorithm development in the first stage. Although the simulator is not able to reproduce the exact behavior of the sand subsurface, some degree of sliding and submergence of the rover can be observed and thus a non-linear behavior can be reproduced. To increase the representativeness and the reliability of the simulation, data from DLR's facilities will be gathered and used to verify the behaviour of the simulated rover in relevant and comparable scenarios. To this end, the rover will ride pre-defined trajectories in DLR's test facility containing a new lunar soil simulant, which is the same as the one that will be used for the new DLR/ESA lunar test facility "LUNA" in Cologne. The resulting trajectory data will be then compared to those from the simulator and simulation parameter will be adjusted accordingly. Furthermore, final tests demonstrating the

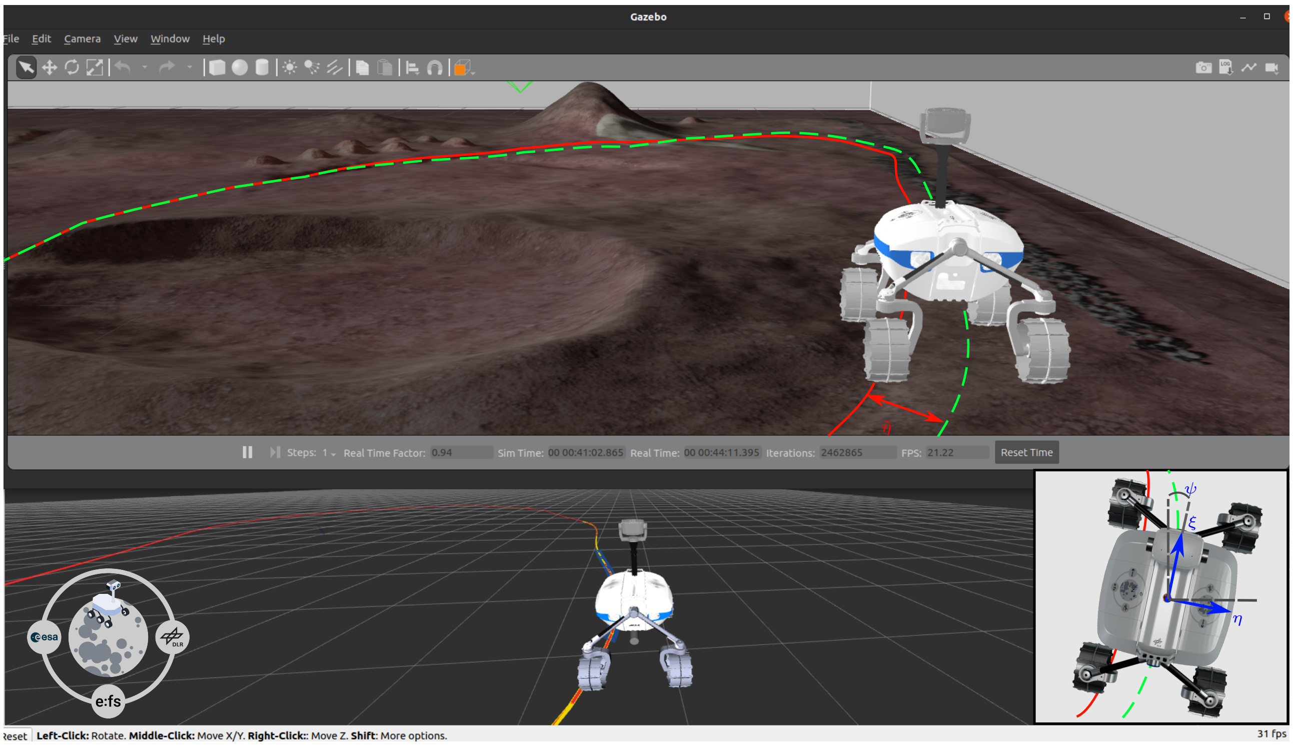
Gazebo
File Edit Camera View Window Help
Steps: 1 Real Time Factor: 0.94 Sim Time: 00 00:41:02.865 Real Time: 00 00:44:11.395 Iterations: 2462865 FPS: 21.22
Reset Time
ψ
ξ
η
esa
e:fs
DLR
Reset Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click:: Move Z. Shift: More options.
31 fps

Figure 6: Gazebo Simulation of the LRU model

functionality and learning properties of the developed algorithms will be performed in the indoor Planetary exploration lab (PEL) of DLR shown in Fig. 5.

4 LEARNING-BASED APPROACHES

Several modules will be developed in different parts of the Rover system during the course of the project. The overall system, as shown in 7, consists of several components, for instance, the rover or the representing simulation model, sensors, a perception module, low-level actuator controller and high-level motion controller. There are two control modules which are designed each to be tested and analysed independently. Furthermore, the algorithms and interfaces are generic to be applicable to both the rover and the simulation.
Static (non-learning) algorithms are implemented for tasks such as trajectory planning and interfacing the wheel actuators and are necessary for testing the closed loop. Also, a non-linear, Lyapunov-based, controller is applied for trajectory tracking control. Besides, three modules with learning-based components are involved. The perception module includes image recognition for segmentation and classification of the surface. The learning-based kinematics adjustments module manipulates the low-level wheel steering controller and the learning-based motion control is applied on a high-level. The perception module is not within the scope of this paper.
Below, first implementations of each of the two latter algorithms are presented and results are discussed.

4.1 Learning-based wheel kinematics adjustment

In this problem setting, it is assumed that a realizable path is given from the planning layer. For this scenario, we will develop a nominal controller, consisting of a nonlinear controller and a static mapping for commanding the individual wheels of the rover (purple blocks in 7). The static mapping

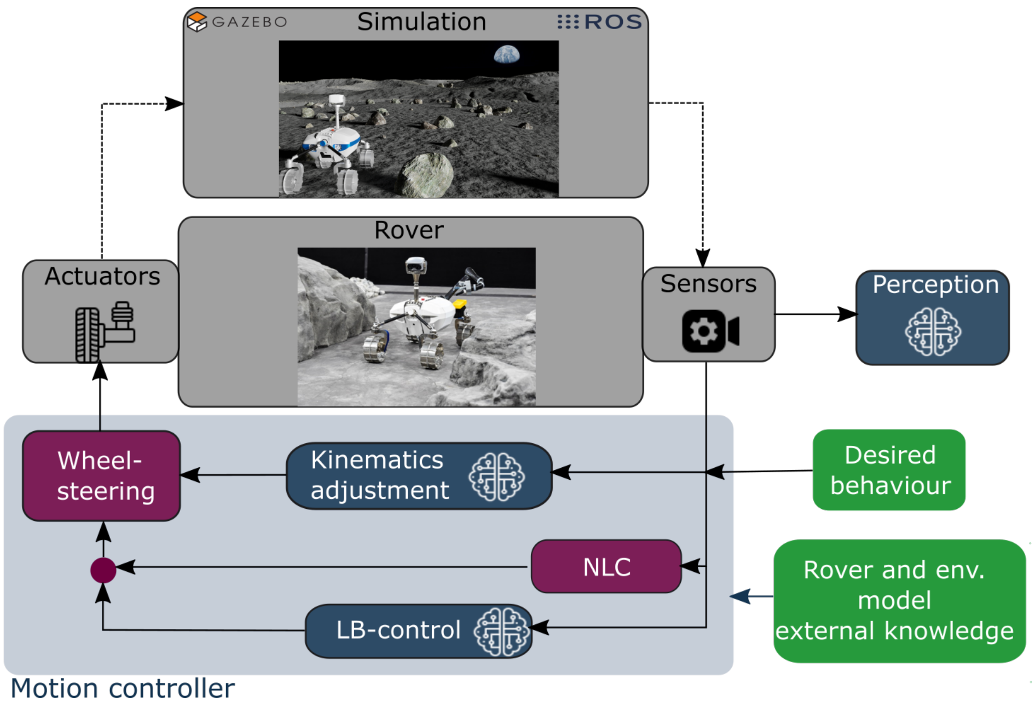

Figure 7: Overall system overview with all project modules

calculates the wheel speeds and steering angle based on the geometrics of the rover and the commanding body yaw rate. These components will enable the rover to track the given trajectory under perfect conditions, that is, when there is no slippage of the rover due to steep or rough terrain. On top of this, a functionality for adjusting the wheel kinematics such that the slippage can be compensated for, will be developed. This functionality needs to ”learn” the required adjustments based on data acquired in the various phases, as described in Section 2 and is the main objective for development. For the learning-based kinematics adjustments module a first algorithm is implemented, which adapts the control output of the nonlinear controller with an additive term. This changes the input of the wheel steering controller and consequently impacts the rover movement. For this purpose, an approximation function is designed, that is online trained on tracking data, to improve the tracking behavior. Due to implementation reasons, we chose to use Gaussian Process Regression (GPR). GPR utilizes a Gaussion Process (GP) [2], [1], model $g(\mathbf{z}) \sim \mathcal{GP}(\mathbf{m}(\mathbf{z}), \mathbf{k}(\mathbf{z}, \mathbf{z}'))$.
To build a regression model by treating the model as a prior, defined only by a covariance function also known as kernel function k. In our case, we use the common squared exponential kernel function. By adding observation data,

$$\mathcal{D} = \{\mathbf{Z} = [\mathrm{z}_1, ..., \mathrm{z}_\mathrm{N}] \in \mathbb{R}^{n_\mathrm{z} \times \mathrm{N}}\ ,\ \mathbf{Y} = [\mathrm{y}_1, ..., \mathrm{y}_\mathrm{N}] \in \mathbb{R}^{1 \times \mathrm{N}}\}, \tag{1a}$$

where n_z denotes the number of feature in the observation vector z and N the number of data points (zk, yk), the GP model can be used as a posterior distribution. This posterior distribution can then be used to predict output values $\mu^d(z^*)$ and its probability $\sum^d(z^*)$ for the desired data points z^* (test points),

$$\mu^\mathrm{d}(\mathbf{z}_*) = \mathbf{K}_*^\top[\mathbf{K} + \sigma_\mathrm{n}^2 \mathrm{I}]^{-1}\mathbf{Y}, \tag{2a}$$
$$\Sigma^\mathrm{d}(\mathbf{z}_*) = \mathbf{K}_{*,*} - \mathbf{K}_*^\top[\mathbf{K} + \sigma_\mathrm{n}^2 \mathrm{I}]^{-1}\mathbf{K}_*, \tag{2b}$$

where d denotes the d-th dimension of the output. K, K^* and $K^{*,*}$ are defined as $k(Z, Z)$, $k(Z, z^*)$ and $k(z^*, z^*)$, respectively.

The feature of the observation vector z in the proposed approach consists of the current trajectory dynamics, described by the curvature and the velocity, and the roll angle of the bogie axles of the rover. The corresponding output (additive yaw rate for the nonlinear controller) of the observation is determined by the current tracking performance, the lateral deviation and heading error, according to the trajectory dynamics. The number of data points N of the observation data is limited, due to increasing computational costs with every data point. The selection of observation data is currently handle with an FIFO method. Other methods, such as covariance based ones, have to be investigated. The hyperparameter of the GPR are continuously optimized via the log marginal likelihood method [2].
Fig. 8 illustrates the improved tracking performance (reduction of lateral deviation) of this approach after two laps. During the first lap, only the collection of observation data is enabled. The additional yaw rate in the second lap, predicted by the GPR, globally improves the tracking performance of the rover. Especially the areas where the lateral deviation exceeds 0,2m are improved significantly.

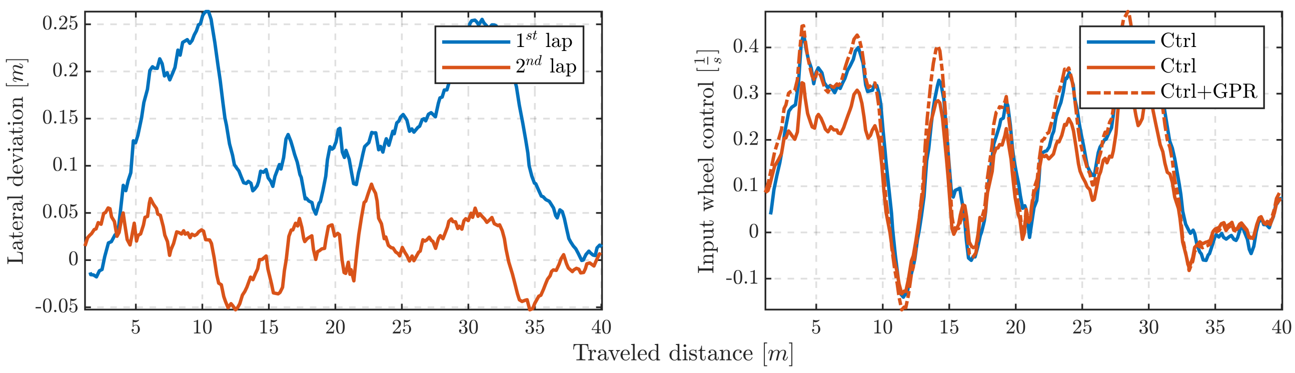

Figure 8: Overall system overview with all project modules

4.2 Learning-based motion control

Also in this problem a realizable trajectory is provided by a planner. We further assume a nonlinear tracking controller and a static wheel steering function that are suited to let the rover track the given trajectory under ideal circumstances. However, in reality, there will be deviations due to unknown aspects in the physical model as well as uncertain interactions with the environment. To reduce these deviations, an iteratively learning model predictive control (MPC) framework will be employed. As main advantage of the MPC formulation, costs and constraints on inputs and the system state can be considered explicitly. The iterative learning formulation allows the controller to improve a dynamical model of the system during operation and with it, the overall system performance. Development for this problem setting will focus on the adaption and implementation of the learning based MPC.
Numerous approaches can be found in the literature that aim to improve the performance and robustness of optimisation-based control approaches such as MPC even in the presence of unknowns and uncertainties. A learning component can be integrated here at various points. For example, through a learning-based model in the form of a robust or stochastic parametric model as in [10] or [8]. However, it is also possible to learn parts of the controller design, such as the cost functions and terminal components, or the entire controller [11].
In out approach, the system is first linearised in combination with a non-linear controller and then controlled with a learning-based MPC. This approach simplifies the optimisation problem and reduces the dependence on an iterative process. To be more specific, the chosen system architecture enables that the mpc only receives the error states of the system, i.e. the deviations from the specified

trajectory.Therefore, it is no longer necessary for the learning algorithm to repeatedly run exactly the same trajectory, as for example in [9].
The general MPC problem for our method can be described as follows:

$$\min_{u,\lambda} \sum_{k=0}^{N-1} J_k + P^{j-1}(\tilde{\boldsymbol{x}}_N), \tag{3}$$

$$P^{j-1}(\tilde{\boldsymbol{x}}_N) = \sum_k \sum_i J_k^i \lambda_k^i, \tag{4}$$

$$J_k = J(\tilde{\boldsymbol{x}}_k) := \tilde{\boldsymbol{x}}_k^T \boldsymbol{Q} \tilde{\boldsymbol{x}}_k + \boldsymbol{u}_k^T \boldsymbol{R} \boldsymbol{u}_k$$

s.t.

$$\tilde{\boldsymbol{x}}_{k+1|j} = \bar{g}(\tilde{\boldsymbol{x}}_{k|j}, \boldsymbol{u}_{k|j}) + g_t^j(\tilde{\boldsymbol{x}}_{k|j}, \boldsymbol{u}_{k|j}) \tag{5}$$

$$\tilde{\boldsymbol{x}}_0 = \tilde{\boldsymbol{x}}(t_0) \tag{6}$$

$$\tilde{\boldsymbol{x}}_k \in \tilde{\mathcal{X}}, \boldsymbol{u}_k \in \mathcal{U}, \tag{7}$$

$$\tilde{\boldsymbol{x}}_N \in \mathcal{CS}^{|-\infty} \tag{8}$$

$$\mathcal{CS}^{|-\infty} = \sum_k \sum_i \tilde{\boldsymbol{x}}_k^i \lambda_k^i, \sum_k \sum_i \lambda_k^i = 1, \lambda_k^i \geq 0 \tag{9}$$

The problem formulation generally consists of a cost function divided into stage costs J_k and terminal costs P. The terminal costs in our case considers the safe sets of past iterations $\mathcal{CS}^{|-\infty}$. Another important part is the description of the system dynamics $\tilde{\boldsymbol{x}}_{k+1|j}$, here divided into a nominal known part $\bar{g}(\tilde{\boldsymbol{x}}_{k|j}, \boldsymbol{u}_{k|j})$ and an unknown part $g_t^j(\tilde{\boldsymbol{x}}_{k|j}, \boldsymbol{u}_{k|j})$, which has to be identified by different methods presented later. In addition, formulations for a constraint of the control input, the error states and the terminal state are given. The Unknown system components are learned either by a parametric model as in [6] or by a deep neural network (DNN).
In the parametric model approach, the deviations of the individual states are determined via a simple last meas square optimization consisting of a feature vector, parameter vector and an error matrix. The feature vector consists of values available in each time step such as the states, the control input and the position of the rover. The error matrix consists of system dynamic errors from a certain number of past iterations and time steps that are very similar to the current state of the rover. These past time steps are searched for and selected using a time index. This approach leads to slight improvements in the simulation.

The DNN used for the approximation and predection of the system dynamics is a Feed Forward Neural Network (FFNN) with ReLU activation features and linear activation features in the output layer. The width of the DNN was three times the number of input features in each layer with a depth of 5 layers. For training the DNN in the Keras framework, datasets with iterations are generated in the Gazebo simulation environment. One iteration consists of a chosen length of 200 time steps. The feature vectors used are the deviations from the reference trajectory, the current and desired velocities, the inclination and pitch of the rover and the last control inputs. The DNN then predicts the expected error in the state prediction for the next time step and is repeated for the whole horizon. For faster prediction and use in optimization, the mesh is stored as a frozen graph, achieving a prediction time of 10 μs for one time step.

First Simulation results show that the proposed controller is able to track a given trajectory while considering state and input constraints and improving performance with the time. Fig. 9 shows the lateral deviation of the rover in the simulation in relation to the reference trajectory. Three different controller configurations are compared. The MPC with nominal model, with the adaptive parametric model and the learning DNN model. For all controllers, the exact same previously trajectory was

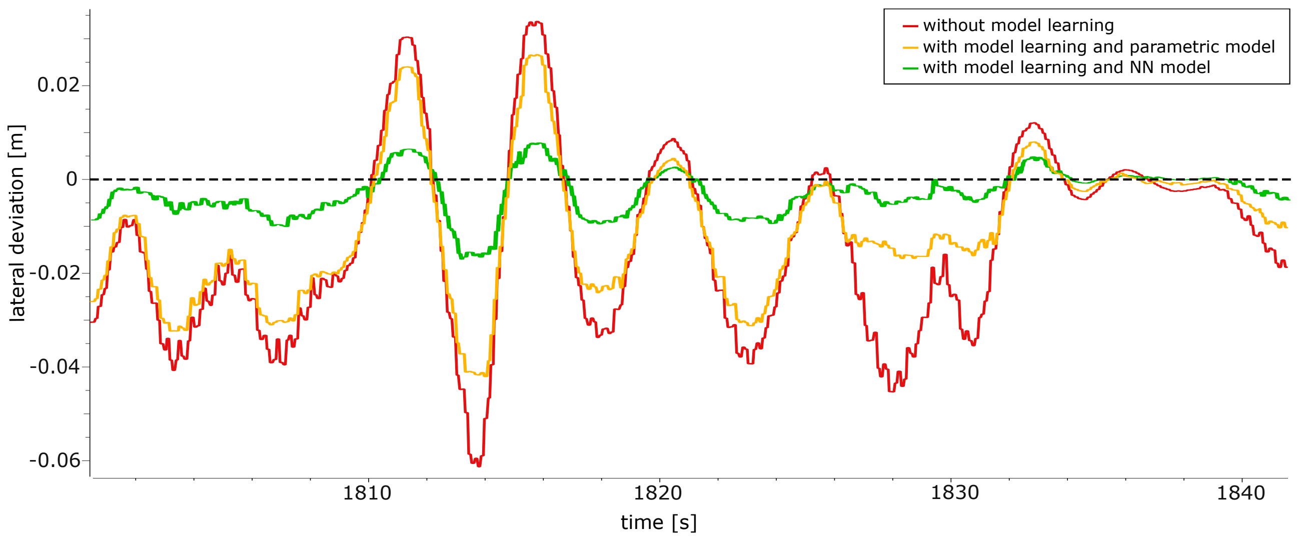

Figure 9: Lateral deviation of the closed loop system with the classic MPC, the LMPC with parametric model and NN model

specified at the same position in the simulation for a better comparison. The ground was slightly hilly and slippery. In the comparison it turns out that the parametric model brings only a slight improvement in contrast to the DNN model. Even if the DNN model already leads to good results, there is still potential for improvement.

5 CONCLUSION

In summary, the current phase of the project has established a solid framework for the development of various learning-based algorithms, including a simulation environment for data generation and testing of the learning-based methods. In addition, the first promising approaches have been developed and implemented. The next steps are to further develop the DNN of the LBMPC using the training data from the validated simulator or real measurement series and embed it in an online learning framework. In addition, the other approaches will be tested and improved on the real rover. Finally, the results will be verified with the rover to compare them with those from the simulation. At the end of the project, a test and demonstration campaign is planned at DLR facilities. From these, not only the improvements achieved by the learning approaches should be derived and presented, but also the hurdles and additional efforts that were necessary for the implementation of the advanced methods should be shown.

REFERENCES

[1] C. E. Rasmussen, "Gaussian processes in machine learning," in *Advanced Lectures on Machine Learning: ML Summer Schools 2003, Canberra, Australia, February 2 - 14, 2003, Tübingen, Germany, August 4 - 16, 2003, Revised Lectures*, O. Bousquet, U. von Luxburg, and G. Rätsch, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 63–71, ISBN: 978-3-540-28650-9. DOI: 10.1007/978-3-540-28650-9_4. [Online]. Available: https://doi.org/10.1007/978-3-540-28650-9_4.

[2] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning* (Adaptive computation and machine learning). MIT Press, 2006, ISBN: 026218253X. DOI: 10.1007/978-3-540-28650-9_4. [Online]. Available: https://www.worldcat.org/oclc/61285753.

[3] A. Wedler, B. Rebele, J. Reill, *et al.*, "Lru – lightweight rover unit," in *Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA)*, Mai 2015. [Online]. Available: https://elib.dlr.de/102155/.

[4] F. Borrelli, A. Bemporad, and M. Morari, *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press, 2017. DOI: 10.1017/9781139061759.

[5] L. Grüne and J. Pannek, *Nonlinear Model Predictive Control*. Springer Cham, 2017, ISBN: 9780857295002.

[6] U. Rosolia, A. Carvalho, and F. Borrelli, "Autonomous racing using learning Model Predictive Control," *Proceedings of the American Control Conference*, pp. 5115–5120, 2017, ISSN: 07431619. DOI: 10.23919/ACC.2017.7963748. arXiv: 1610.06534.

[7] M. J. Schuster, S. G. Brunner, K. Bussmann, *et al.*, "Towards Autonomous Planetary Exploration: The Lightweight Rover Unit (LRU), its Success in the SpaceBotCamp Challenge, and Beyond," *Journal of Intelligent & Robotic Systems (JINT)*, Nov. 2017, ISSN: 1573-0409.

[8] R. Soloperto, M. A. Müller, S. Trimpe, and F. Allgöwer, "Learning-Based Robust Model Predictive Control with State-Dependent Uncertainty," *IFAC-PapersOnLine*, vol. 51, no. 20, pp. 442–447, 2018, ISSN: 24058963. DOI: 10.1016/j.ifacol.2018.11.052.

[9] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger, "Learning-Based Model Predictive Control for Autonomous Racing," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3363–3370, Oct. 2019, ISSN: 2377-3766. DOI: 10.1109/LRA.2019.2926677. [Online]. Available: https://ieeexplore.ieee.org/document/8754713/.

[10] M. Lorenzen, M. Cannon, and F. Allgöwer, "Robust MPC with recursive model update," *Automatica*, vol. 103, pp. 461–471, 2019, ISSN: 00051098. DOI: 10.1016/j.automatica.2019.02.023.

[11] S. Gros and M. Zanon, "Data-driven economic nmpc using reinforcement learning," *IEEE Transactions on Automatic Control*, vol. 65, no. 2, pp. 636–648, 2020. DOI: 10.1109/TAC.2019.2913768.

[12] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, "Learning-Based Model Predictive Control: Toward Safe Learning in Control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, pp. 269–296, 2020, ISSN: 2573-5144. DOI: 10.1146/annurev-control-090419-075625.

[13] M. J. Schuster, M. G. Müller, S. G. Brunner, *et al.*, "The arches space-analogue demonstration mission: Towards heterogeneous teams of autonomous robots for collaborative scientific sampling in planetary exploration," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5315–5322, 2020. DOI: 10.1109/LRA.2020.3007468.

[14] A. Mesbah, K. P. Wabersich, A. P. Schoellig, *et al.*, "Fusion of Machine Learning and MPC under Uncertainty: What Advances Are on the Horizon?" *Proceedings of the American Control Conference*, vol. 2022-June, no. 1, pp. 342–357, 2022, ISSN: 07431619. DOI: 10.23919/ACC53348.2022.9867643.