MODEL PREDICTIVE TRACTION AND STEERING CONTROL OF PLANETARY ROVERS

Rainer Krenn[(1)], Andreas Gibbesch[(1)], Giovanni Binet[(2)], Alberto Bemporad[(3)]

[(1)] DLR, RMC; Oberpfaffenhofen, 82234 Wessling, Germany; {rainer.krenn | andreas.gibbesch}@dlr.de
[(2)] GMV; c/Isaac Newton, 1; 28027 Tres Cantos, Spain; gbinet@gmv.com
[(3)] IMT; Piazza S. Ponziano, 6, 55100 Lucca, Italy; alberto.bemporad@imtlucca.it

ABSTRACT

Results of the ESA project RobMPC (Robust Model Predictive Control for Space Constraint Systems) could successfully demonstrate that model predictive control (MPC) is definitively applicable for space systems with high dynamics like wheeled vehicles exploring a planetary surface. In the context of RobMPC a rover control hierarchy for guidance, trajectory control as well as traction and steering control was implemented. Controller verifications and robustness tests were performed using a functional engineering simulator (FES) including a multi-body dynamics model of ESA's EGP rover and the vehicle-terrain contact physics. The latest validation step is the MPC implementation on a real-time computer system controlling the ExoMars breadboard rover at DLR's planetary exploration lab.

1 INTRODUCTION

Within space applications a typical domain of model predictive control (MPC) is satellite attitude and orbit control, e.g. focused in ESA's ORCSAT project [1]. Since satellite systems have just moderate dynamics and their operational environment is more or less well known, predictive control algorithms relying on numerical models appear to be promising solutions.

In case of rover control the dynamics conditions are much more complicate. The reasons are the multi-body kinematics and dynamics of the vehicle chassis, the complex contact dynamics between wheels and soil and the big uncertainties regarding contact dynamics parameters, in particular when driving long distances in unknown planetary terrain. Accordingly, the extension of MPC solutions to rover applications seems to be by no means trivial. Nevertheless, the major advantages of MPC like

- the capability of controlling multi-input and over-actuated systems,
- the integrated, stringent handling of constraints and
- the option of optimizing the control output regarding user-defined goals

make the method still very attractive for application in the context of high dynamics systems.

MPC control solutions for planetary rovers were recently investigated within the RobMPC project (Robust Model Predictive Control for Space Constraint Systems) under ESA contract (see Section ACKNOWLEDGEMENT). In this project the MPC approach was applied to three layers of a rover control hierarchy, which are representing medium to high dynamics control tasks:

1. Guidance,
2. Trajectory control and
3. Wheel traction and steering control (TSC).

In order to avoid the need of being expert on both, rover system dynamics and MPC control theory, a novel MATLAB/Simulink based toolbox called MPCSofT was developed by IMT within the RobMPC project. It provides an environment for design and verification of MPC controllers, based on a quite general class of linear time-varying models, constraints and quadratic costs, possibly equipped with integral action to increase robustness. Provided that the user model is formulated as a linear state space model, MPCSofT is able to access the model for state prediction and control input optimization within the MPC control process.

The scope of RobMPC was to assess the performances of MPC controllers at different levels of rover locomotion control and benchmarking them against classic controller solutions. In addition, the objective was to validate the robustness of the designed controllers by varying the vehicle and environment parameters outside their nominal values as applied in the MPC's prediction model.

In the following the paper will give a short introduction in implementation and utilization aspects of MPCSofT in Section 2. Section 3 will briefly present controller hierarchy and test results from the overall planetary rover application scenario within RobMPC. The focus of the paper will be on wheel traction and steering control, introduced in Section 4. This part includes details of the prediction model, the implementation of the controller and results of performance tests. Here, the section refers to both, verification using a functional engineering simulator and validation with a real system controlled in real-time.

2 MPC IMPLEMENTATION

Model Predictive Control (MPC) is an advanced control technique that explicitly uses

1. a dynamics model of the process to predict its evolution over a given time horizon in the future and
2. numerical optimization methods to determine the optimal sequence of control inputs that minimizes a given performance index under constraints on control inputs and state variables.

The prediction and optimization is repeated at each sampling step. Thus the applied optimal control input is always based on the latest available feedback from sensors or observers.

The MATLAB/Simulink toolbox MPCSofT is designed for control of linear time-varying (LTV) systems, which are to be described in the user's prediction model by three equations for

1. the linear, time-varying system dynamics (1),
2. time-varying cost function (2) and
3. time-varying constraint functions (3).

$$\mathbf{x}_{k+1} = \mathbf{A}(t,k,\mathbf{x}(t))\mathbf{x}_k + \mathbf{B}(t,k,\mathbf{x}(t))\mathbf{u}_k + \mathbf{f} \quad (1)$$

$$\mathbf{z}_k = \mathbf{E}_z(t,k,\mathbf{x}(t))\mathbf{x}_k + \mathbf{H}_z(t,k,\mathbf{x}(t))\mathbf{u}_k + \mathbf{P}_z(t,k,\mathbf{x}(t))\Delta\mathbf{u}_k \quad (2)$$

$$\mathbf{c}_k = \mathbf{E}_c(t,k,\mathbf{x}(t))\mathbf{x}_k + \mathbf{H}_c(t,k,\mathbf{x}(t))\mathbf{u}_k + \mathbf{P}_c(t,k,\mathbf{x}(t))\Delta\mathbf{u}_k \quad (3)$$

The MPC performance index to be minimized at each sampling step is

$$\rho_1\varepsilon_1 + \rho_2\varepsilon_2 + \sum_{k=0}^{N(t)-1}(\mathbf{z}_k - \mathbf{r}_k)^T(\mathbf{z}_k - \mathbf{r}_k), \quad (4)$$

subject to three types of constraints, in particular

1. inequality constraints on states and on control inputs (5),
2. terminal equality constraints on states at end of prediction (6) and
3. equality constraints on the input increment for prediction steps beyond the control horizon (7).

$$\mathbf{c}_k \le \mathbf{c}_{max}(t,k,\mathbf{x}(t)) + \mathbf{V}_c(t,k,\mathbf{x}(t))\varepsilon_1; \quad 0 \le k < N(t) \quad (5)$$

$$\mathbf{C}_N(t,\mathbf{x}(t))\mathbf{x}_{N(t)} \le \mathbf{d}_N(t,\mathbf{x}(t)) + \mathbf{V}_N(t,\mathbf{x}(t))\varepsilon_2 \quad (6)$$

$$\Delta\mathbf{u}_k = \mathbf{0}, \quad N_U(t) \le k < N(t) \quad (7)$$

The variables used in (1) - (7) are listed in Tab. 1. They are grouped by input and output arguments of the user's prediction model function and by variables internally processed by MPCSofT.

Equations (1) - (7) formulate a finite-time optimal control problem that is mapped into a quadratic programming (QP) problem, in which the optimization variables are

- the slack variables ε, numerically required for handling unrealizable constraints and
- the sequence of control input increments ($\Delta\mathbf{u}_0$, $\Delta\mathbf{u}_1$, … , $\Delta\mathbf{u}_N$) applied to the dynamics model over the prediction horizon.

The first optimal control input increment $\Delta\mathbf{u}_0$ is finally selected to be applied to the actual controlled system.

Variable Name	Meaning
Prediction model input arguments from MPCSofT	
t	Current time at start of prediction
T_s	Controller sampling step size
k	Current number of prediction step
N, N_u	Prediction horizon, control horizon
$\mathbf{x}(t)$	Initial state at beginning of prediction
Prediction model output arguments to MPCSofT	
$\mathbf{A}$, $\mathbf{B}$, $\mathbf{f}$	State space matrices and vectors
$\mathbf{E}_z$, $\mathbf{H}_z$, $\mathbf{P}_z$	Performance description matrices
$\mathbf{E}_c$, $\mathbf{H}_c$, $\mathbf{P}_c$	Constraint description matrices
$\mathbf{C}_N$	Terminal constraint description matrix
$\mathbf{r}$	Performance reference vector
$\mathbf{c}_{max}$	Constraint reference vector
$\mathbf{d}_N$	Terminal constraint reference vector
$\mathbf{V}_c \ge 0$, $\mathbf{V}_N \ge 0$	Vectors for constraint hardening
ρ_1, ρ_2	Weights on constraint slack variables
Internal controller variables	
$\mathbf{x}$	State vector of system
$\mathbf{u}$	Control input vector
$\mathbf{z}$	Performance vector to be optimized
$\mathbf{c}$	Constraint vector
$\Delta\mathbf{u}_k = \mathbf{u}_k - \mathbf{u}_{k-1}$	Control input increment vector
ε_1, ε_2	Slack variables for constraint softening

Table 1: Variables Used for Prediction Model

The user can exploit the maximum flexibility offered by the Embedded MATLAB (EML) language to define the prediction model according to (1) - (3) in an EML module. This module is used inside the LTV-MPC Simulink block of the MPCSofT toolbox, shown in Fig. 1.

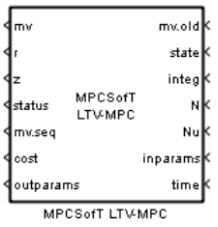

Figure 1. LTV-MPC Simulink block

In fact, the block contains

1. a QP builder function called once for mapping of the optimal control problem at initialization and
2. a QP solver function repeatedly called at each sampling step for optimization of the control inputs.

As the block is completely designed based on Simulink blocks and EML functions, C code can be immediately generated for rapid prototyping.

3 MPC FOR ROVER CONTROL HIERARCHY

The mission proposed in the rover locomotion (ROL) scenario of RobMPC is a human type mission, primarily fulfilling human-related tasks like transporting soil, relocating cargo, inspecting the landing site for human arrival or the deployed infrastructure. In this context, the Eurobot Ground Prototype (EGP) rover has been selected as the targeted platform (see Fig. 2 and [3]). The main characteristic parameters of the EGP rover are collected in Tab. 2.

Parameter	Value
Bounding box volume (L/W/H)	2246/1580/1505 mm
Total weight	880 kg
Track width (front/rear)	1041/1330 mm
Axle distance	1511 mm
Steering axle	rear wheel steering
Maximum velocity	0.35 m/s
Navigation sensor	IMU

Table 2: EGP rover parameters

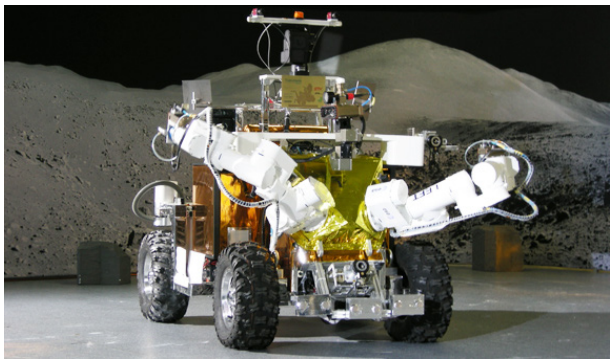

Figure 2: EGP rover

Concerning the environment in which the rover has been tested, a Martian scenario has been chosen. The soil is supposed to have negligible cohesion, which is one of the major differences compared with terrestrial off-road conditions. A playground of 20 x 20 m with possibly inclined topology and a number of obstacles has been chosen, together with a nominal path to be followed to test the controllers. The path is off-line computed by a modified A* algorithm. It is composed by at least two turns and a straight part and takes already into account the steering capabilities of the vehicle that has to avoid.

The selected rover control hierarchy consists of three MPC controllers designed within the RobMPC project (see Fig. 3):

1. Guidance function, which acts as an online path planner by continuously computing an obstacle-free optimal contingency path which, when the vehicle gets outside a safety corridor around the nominal path, re-injects the rover back into it.
2. Trajectory control, which computes the desired rover velocity vector to be followed based on the current position and orientation with respect to the path to be followed.
3. The wheel traction and steering controller, which commands wheel actuators with the desired steering angle and wheel velocity based on the desired rover velocity vector under consideration of the terrain conditions.

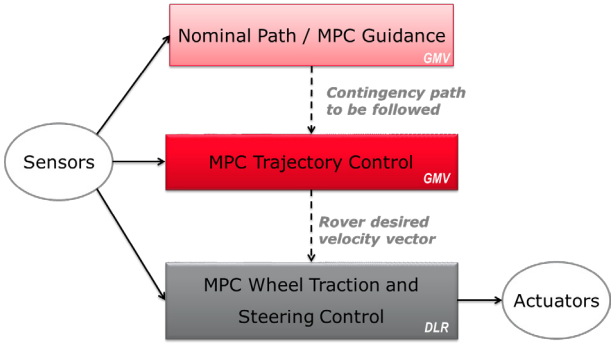

Figure 3: MPC controllers

The smooth interaction, when all three controllers are working together in the loop has been successfully proven by extensive integration tests using a functional engineering simulator (FES). It includes the following major components for adequately realistic simulation of the test scenario:

- Multi-body dynamics model of the EGP-rover including wheel and steering actuator dynamics,
- Models of the sensors required for control feedback, in particular an IMU and a position sensor,
- 3D terrain topology model,
- 3D / 6DOF soil contact dynamics model based on DLR's soil contact model tool SCM [4].

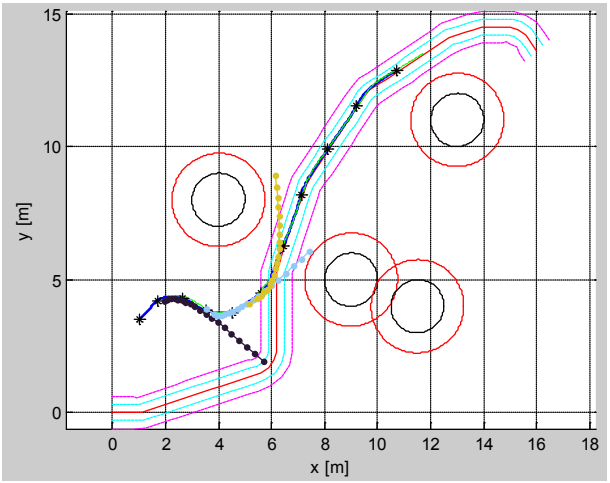

Figure 4: Results with all MPC controllers working together in the rover control hierarchy

The controller performance was tested by letting the rover start in a position far away from the nominal path, and then let the MPC guidance continuously compute contingency paths to bring the rover back to the nominal path in an optimal way without colliding with obstacles. The realization of the desired path was finally performed by the subordinate MPC controller. A sample result obtained is shown in Fig. 4. For more details on implementation, performance and robustness tests of MPC guidance and MPC trajectory control developed in the context of RobMPC please refer to [5].

4 TRACTION AND STEERING CONTROL

In this section the focus will be on MPC wheel traction and steering control (TSC). It is the most vehicle specific layer in the control hierarchy (Fig. 3) and is introduced by means of two different rover systems:

1. The EGP rover (see Tab. 2 and Fig. 2) in form of a numerical model as implemented in the FES used for RobMPC and
2. The ExoMars breadboard rover (Fig. 5 and [6]), physically available in the planetary exploration testbed (PEL) at DLR and utilized for additional RobMPC verification and validation activities. The rover parameters are summarized in Tab. 3.

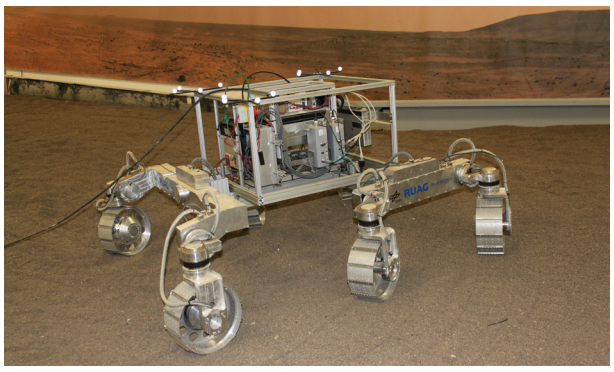

Figure 5: ExoMars breadboard rover at DLR's PEL

Parameter	Value
Bounding box volume (L/W/H)	1600/1370/~800 mm
Total weight	90 kg
Track width (front/center/rear)	1200/1200/1200 mm
Axle distance (front-center-rear)	640/720 mm
Steering axle	all wheel steering
Maximum velocity	0.03 m/s
Navigation sensor	Camera based 3D pose tracking

Table 3: ExoMars rover breadboard parameters

TSC is the interface between the trajectory controllers and the low level, device specific controllers for individual control of wheel and steering actuators (see Fig. 6). It is a coordinating control instance that has to have detailed knowledge of the vehicle kinematics and replaces typically conventional Ackermann control. Its location inside the control hierarchy is equivalent to ABS and ESP known from road vehicles.

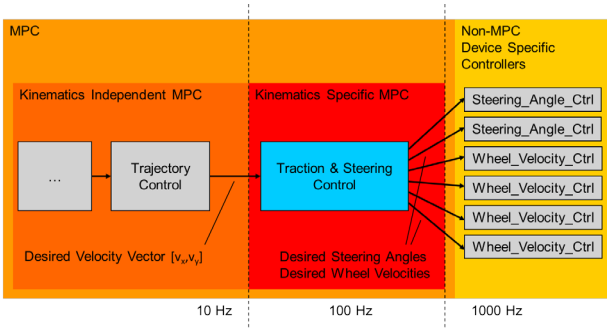

Figure 6: TSC location inside rover control hierarchy

Accordingly, the controller computes desired wheel velocities $\boldsymbol{\omega}$ and steering reference angles $\boldsymbol{\beta}$, which define the control input vector $\mathbf{u}$, in order to realize the desired trajectory given by the sequence of velocity vectors $\mathbf{v}_{traj}$ with two linear components each. However, the rover system has three states $\mathbf{x}$, which are the longitudinal and lateral translational vehicle velocities as well as the angular rate around the vertical vehicle axis. Thus, the system is under-determined. In particular, the heading angle is freely selectable giving TSC the opportunity to let the rover drift and slide, which is an important dynamic feature for off-road vehicle locomotion performance. Moreover, the system is over-actuated providing the opportunity for taking additional control goals into account, beyond following the desired trajectory.

The MPC prediction horizon N is defined by the sampling step size ratio of TSC and its subordinated actuator controllers (see reference values in Fig. 6). The typical value is 10.

The implementation of MPC-TSC utilizing MPCSofT (see Section 2) is introduced in the following subsections. The procedures of the controller design are identical for both, EGP rover and ExoMars rover.

4.1 System Dynamics

The system dynamics of the rover is represented in the prediction model by a double-track vehicle model. It is able to take different dynamics at left and right wheels into account, which is crucial for cross-hill locomotion control. Vertical system dynamics is neglected within the double-track model. Corresponding to the rover the model has three degrees of freedom and can perform motions on a possibly inclined plane. Thus, one obtains the vehicle's equations of motion, (w.r.t. the vehicle's center of mass, expressed in vehicle reference frame coordinates) as function of gravity and wheel forces as given in (8). The variables used in (8) are listed in Tab. 4.

$$\left.\begin{aligned} m\dot{v}_{long} &= mg_{long} + \sum_{i=1}^{n_{Wheel}} F_{long,i} \\ m\dot{v}_{lat} &= mg_{lat} + \sum_{i=1}^{n_{Wheel}} F_{lat,i} \\ J_{zz}\dot{\omega}_z &= \sum_{i=1}^{n_{Wheel}} \left(r_{long,i}F_{lat,i} + r_{lat,i}F_{long,i} + T_{z,i} \right) \end{aligned}\right\} \quad (8)$$

Variable Name	Meaning
$v_{long}, v_{lat}, \omega_z$	Vehicle states: Longitudinal and lateral velocity, angular rate w.r.t. vertical axis
m, J_z	Mass of rover, moment of inertia w.r.t. center of mass around vertical axis
g_{long}, g_{lat}	Longitudinal and lateral components of gravitational acceleration
i, n_{Wheel}	Wheel index, number of wheels
$F_{long,i}, F_{lat,i}, T_{z,i}$	Forces / torques due to wheel-soil dynamics, applied to wheel center: Longitudinal and lateral forces, torque w.r.t. vertical vehicle axis
$r_{long,i}, r_{lat,i}$	Longitudinal and lateral wheel center locations coordinates

Table 4: Variables used in rover's equations of motion

The two forces and the torque referenced in (8) are components of the soil contact forces and torques, mapped onto the wheel center. They are integrals over the wheel-soil contact zone computed as functions of the specific contact force

$$\mathbf{f} = \sigma \mathbf{n}_0 + \tau \mathbf{t}_0 , \quad (9)$$

with σ and τ denoting normal and shear stress and $\mathbf{n}_0$ and $\mathbf{t}_0$ denoting their direction of application.

The key equations for computing normal and shear stress are the well-known pressure-sinkage relationship for soft soil proposed by Bekker [7],

$$\sigma = \left(\frac{k_c}{b} + k_\varphi \right) z^n , \quad (10)$$

the soil failure approximated by the Mohr-Coulomb failure criterion,

$$\tau_{max} = c + \sigma \tan\varphi \quad (11)$$

and the formulation for shear stress according to Janosi and Hanamoto [8],

$$\tau = \tau_{max} \left(1 - e^{-j/k_j} \right). \quad (12)$$

They are functions of sinkage z, of shear deformation j and of the parameters collected in Tab. 5.

While shear deformation j, applied in (12), can be explicitly approximated by kinematics relationships of current vehicle states **x** and control inputs **u**, the solution for sinkage z, applied in (10), encounters two problems to be solved accordingly:

1. The wheel load distribution at any vehicle inclination: Vehicles with rigid chassis and more than three wheels are statically over-estimated. A solution for the static equilibrium, respectively for the load distribution across the wheels, can be found by applying virtual movements according to the principle of virtual work.
2. The explicit solution of sinkage z as function of the wheel load: Since the relationship of sinkage and load is only implicitly given by the inverse function of (10), a look-up table is computed a-priori and approximated by a polynomial, which is used for explicit solution of sinkage during control.

Variable Name	Meaning
k_c, k_φ, k_j	Cohesive modulus , frictional modulus and deformation modulus of soil
c, φ	Cohesion and internal friction angle of soil
n	Sinkage exponent of soil
b	Soil contact patch width

Table 5: Parameters of wheel-soil contact dynamics

Now all major computational steps are introduced for solving the equations of motion (8). However, due to the complicated solution algorithm for wheel-soil contact forces, the equations are highly non-linear. In order to finally provide the state space matrices **A** and **B**, as required in (1), the system is numerically linearized around the current vehicle state **x**(t) and the current control inputs **u**(t) at each sampling step. **A** and **B** are kept constant during prediction. Thus, the prediction model can be characterized as a step-wise linear time-invariant (LTI) model.

4.2 Cost Function

For optimization of the control inputs **u** with (4) the TSC prediction model provides two reference goals **r**. The primary goal $\mathbf{r}_{pri}$ is optimal tracking of the desired velocity vector $\mathbf{v}_{traj}$ by the vehicle reference point. Since under off-road conditions the desired velocity may be not achievable the goal is intentionally not formulated as a constraint. The secondary goal $\mathbf{r}_{sec}$ considers the internal configuration of wheel velocities and steering angles. In order to obtain energy-optimal solutions, control input configurations $\mathbf{u}_{roll}$ are preferred, where all wheels are purely rolling. This condition is fulfilled when the instantaneous centers of rotation of all wheels coincide. A global goal like this will also protect the control from ending-up in a sub-optimal local minimum in the course of the vehicle operation. The complete performance reference vector is given in (13):

$$\mathbf{r} = \begin{pmatrix} \mathbf{r}_{pri} \\ \mathbf{r}_{sec} \end{pmatrix} = \begin{pmatrix} \mathbf{W}_{pri}\mathbf{v}_{traj} \\ \mathbf{W}_{sec}\mathbf{u}_{roll} \end{pmatrix}. \quad (13)$$

The matrices $\mathbf{W}_{pri}$ and $\mathbf{W}_{sec}$ are used for tuning the MPC controller in terms of weighting the particular optimization goals.

4.3 Constraints

The constraints used in the prediction model are physical-mechanical limits of the wheel and steering actuators. They are partly static and partly dynamic:

1. Steering angle limits: Static due to mechanical steering angle end stops.

$$\boldsymbol{\beta}_{min} \le \boldsymbol{\beta} \le \boldsymbol{\beta}_{max}. \quad (14)$$

2. Steering angle rate limits: Dynamic function of current steering torques $\mathbf{T}_\beta$.

$$\dot{\boldsymbol{\beta}}_{min}(\mathbf{T}_\beta) \le \dot{\boldsymbol{\beta}} \le \dot{\boldsymbol{\beta}}_{max}(\mathbf{T}_\beta). \quad (15)$$

3. Angular velocity limits of wheel actuators: Dynamic function of current wheel torques $\mathbf{T}_\omega$.

$$\boldsymbol{\omega}_{min}(\mathbf{T}_\omega) \le \boldsymbol{\omega} \le \boldsymbol{\omega}_{max}(\mathbf{T}_\omega). \quad (16)$$

4. Angular acceleration limits of wheel actuators: Dynamic function of current angular velocities $\boldsymbol{\omega}$ and current wheel torques $\mathbf{T}_\omega$.

$$\dot{\boldsymbol{\omega}}_{min}(\mathbf{T}_\omega, \boldsymbol{\omega}) \le \dot{\boldsymbol{\omega}} \le \dot{\boldsymbol{\omega}}_{max}(\mathbf{T}_\omega, \boldsymbol{\omega}). \quad (17)$$

With (14) - (17) one can formulate the constraint reference vector $\mathbf{c}_{max}$ for inequality constraints according to (5) as follows:

$$\mathbf{c}_{max} = \left(+\boldsymbol{\beta}_{max}, -\boldsymbol{\beta}_{min}, +\dot{\boldsymbol{\beta}}_{max}, -\dot{\boldsymbol{\beta}}_{min}, +\boldsymbol{\omega}_{max}, -\boldsymbol{\omega}_{min}, +\dot{\boldsymbol{\omega}}_{max}, -\dot{\boldsymbol{\omega}}_{min}\right)^T. \quad (18)$$

Terminal constraints are not considered in TSC.

4.4 Verification and Robustness Tests

Within RobMPC the MPC controller evaluation work was performed using the high fidelity functional engineering simulator (FES) introduced in Section 3.

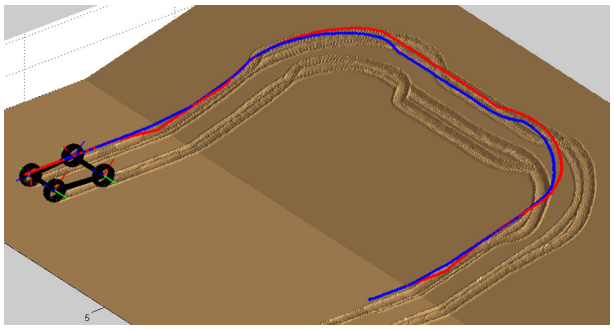

Figure 7: Test Terrain Used for Controller Evaluation

For evaluation purposes a ramp-like terrain and a U-shaped test trajectory (Fig. 7) to be tracked by the rover at constant velocity was selected. The operation takes 90 seconds. The scenario includes all major aspects for controller performance tests:

- Uphill / downhill locomotion,
- Cross-hill locomotion,
- Sharp turns at inclined terrains.

The evaluation was split in two parts: Nominal tests and robustness tests.

Parameter	Nominal	Min	Max
Vehicle Parameters			
Total rover mass	880 kg	810	950
4x wheel mounting misalignment	0°	-0.5°	0.5°
2x steering drive time constant	0.19 s	0.17	0.21
4x wheel drive time constant	0.3 s	0.1	0.59
Terrain Parameters			
Terrain inclination	0°	0°	15°
Environmental gravity	3.71 m/s^2	3.24	4.08
Frictional modulus	1.0e7	1.0e6	1.0e7
Cohesive modulus	0	0	1.0e4
Shear def. modulus	0.1	0.1	0.5
Exponent of sinkage	1	1	1.2
Cohesion of soil	0 Pa	0	30
Internal friction angle	30°	20°	35°

Table 6: Relevant Test Parameters and Variation Range

4.4.1 Results of Nominal Test

The nominal tests were performed with identical parameter values (see Tab. 6) applied in both, the prediction model and in the FES models. Under these conditions the MPC performance was compared with the performance of the reference controllers. In case of TSC the references controllers are

1. Ackermann feedforward control and
2. Ackermann control with PID type velocity and heading angle feedback control.

The performance was evaluated based on a number of performance indicators considering control accuracy, control effort or smoothness of control action. Amongst them the integral of the square error (ISE) over time,

$$\int_t \left(\begin{pmatrix} v_{long} \\ v_{lat} \end{pmatrix}(t) - \mathbf{v}_{traj}(t) \right)^T \left(\begin{pmatrix} v_{long} \\ v_{lat} \end{pmatrix}(t) - \mathbf{v}_{traj}(t) \right) dt, \quad (19)$$

is supposed to be the most meaningful one. The results of (19) for MPC and reference controllers are presented in Fig. 8. It is obvious that MPC can significantly improve the control performance compared to conventional open-loop Ackermann control. But MPC seems to be also the better choice compared to advanced Ackermann control extended by velocity and heading angle feedback control. The reason is the actual number of controls, which is always two in case of Ackermann control (trajectory radius and velocity) but six in case of MPC control of the EGP rover (2 steering angles, 4

wheel velocities). MPC successfully exploits the over-actuation potential of the vehicle by the optimized steering angle - wheel velocity coordination, which is not rigidly linked by kinematic relationships.

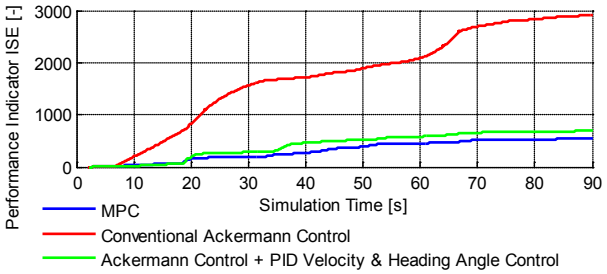

Figure 8: Evolution of Control Performance Indicators (small values indicate small errors)

4.4.2 Results of Robustness Tests

For evaluation of controller robustness regarding uncertainties of

- vehicle parameters and
- terrain parameters

a number of Monte Carlo simulations were performed. In each Monte Carlo simulation campaign one group of parameters was randomly varied and changed in the FES models, while keeping controller parameters constant at nominal values. The variation ranges are documented in Tab. 6.

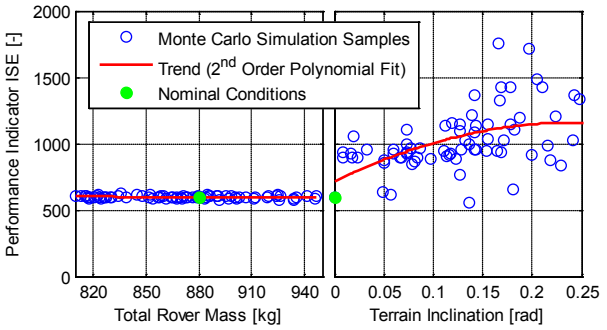

Figure 9: Sensitivity w.r.t. most important parameters

According to Fig. 9, the controller robustness could be successfully proven. The circular markers in the diagrams show the performance indicator values at the end of each simulation run. The values are sorted regarding one parameter of the parameter group in order to identify potential sensitivity trend. In Fig. 9, they are sorted regarding vehicle mass and terrain inclination. According to the diagrams, variations of vehicle parameters have just negligible influence on the MPC performance. Variations of terrain parameters have moderate influence on controller performance even if no clear trend regarding one single parameter could be identified. Nevertheless, in any test cases the MPC performance is still much better than the reference performance of conventional Ackermann control (see Fig. 8).

4.5 Validation with Real System

Within RobMPC the computation of the TSC algorithm in real-time was identified as a critical issue. Moreover, potential inaccuracies of the FES soil model could also reduce the control performance under real terrain conditions. Therefore, validation tests with real-time vehicle control hardware were performed using the ExoMars breadboard rover as target system. The onboard computer parameters are listed in Tab. 7.

Parameter	Value
CPU	INTEL Core2Duo T9600, 2.8GHz
RAM	4 GB, 1066 MHz
OS	QNX 6.5 Neutrino

Table 7: Onboard Computer Parameters

The planetary exploration testbed at DLR, where the rover is operated in, provides a sandbox of 10 x 5 m filled with approximately cohesion-less gravel of Eifel lava. One part of the terrain can be motorized inclined in order to obtain similar conditions as given in the virtual terrain of Fig. 7. The soil parameters are supposed to be inside the variation range defined in Tab. 6. The sensor system used for feedback generation is a vision based tracking system.

A sample result is given in Fig. 10. It shows the tracking of a U-shaped reference trajectory with the ExoMars breadboard rover controlled by MPC. According to 4 the reference trajectory is given as a sampled sequence of desired velocity vectors $\mathbf{v}_{traj}$ over time. The trajectory includes flat, up-hill/down-hill and cross-hill sections for linear motion but also sections for turning on inclined terrain. The result demonstrates that the MPC algorithm, working actually in the velocity domain, is even precise enough for providing good performance in the position domain without position feedback.

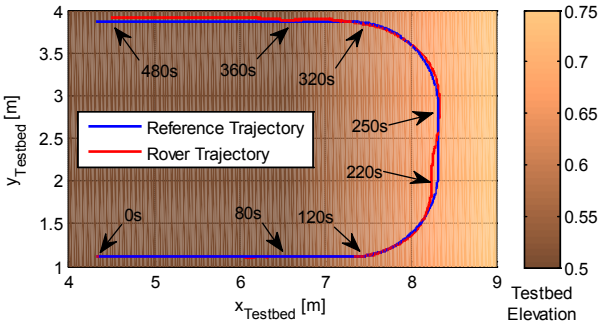

Figure 10: Tracking of U-Shaped Trajectory by MPC

Since the MPC algorithm for 12 independent control inputs (6 steering angles, 6 wheel velocities) could not yet convincingly be solved at the desired sampling rate of 100 Hz the algorithm was slightly changed. In the presented version the control inputs are linked according to Geometry Based Control [9] with mapping

of control input constraints (18) by Jacobians. Thus the number of controls is reduced to 3 and the control goals of (13) are implicitly achieved. This control setup is equivalent to interactive vehicle control using e.g. a side-stick for longitudinal, lateral and turning motion commands. The actual MPC computed control inputs are shown in Fig. 11. The dotted lines show the reference signals to be applied under ideal tracking conditions (no control error, **u = x)**.

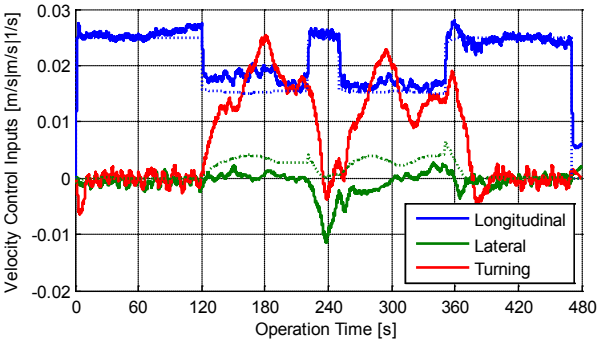

Figure 11: MPC Control Inputs

Even if a direct correlation between simulation results and experimental test results is not possible due to significant differences of the investigated rover systems, one can state that the results regarding control performance obtained by simulations (see 4.4) could be confirmed by applications with real rover hardware in realistic environment.

5 CONCLUSION

In the project RobMPC it could be convincingly demonstrated that MPC is a promising control solution for all layers of the rover control hierarchy: Guidance, trajectory control and wheel traction and steering control. The toolbox MPCSofT, developed within RobMPC, efficiently supports the implementation of LTI and LTV prediction models as well as the user-friendly definition of optimization goals and constraints. Moreover, rapid prototyping can be easily performed with MPCSofT due to the convention of model implementation using Embedded MATLAB code.

Traction and steering control (TSC) plays a key role at the interface of trajectory and actuator controllers and is in the focus of the paper. The presented MPC solution for TSC is based on a step-wise LTI prediction model. It is implemented as a double-track model considering wheel-soil contact dynamics according to Bekker's semi-empirical terramechanics theory. The actuator performance limitations are described as MPC constraints. The MPC control approach could be verified with both, an EGP rover locomotion simulator, mainly used for extensive controller robustness tests, and the ExoMars breadboard rover operated with real-time control and feedback of real tracking sensors.

ACKNOWLEDGEMENT

RobMPC is an ESA/ESTEC project (ESTEC/ITT AO/1-5979/09/NL/JK) completed in 2012. The technical work was supervised by Samir Bennani and Eric Bornschlegl with consulting from Michel van Winnendael and staff from ESTEC robotics section.

REFERENCES

[1] Saponara, M., Barrena, V., Bemporad, A., Hartley, E.N., Maciejowski, J., Richards, A., Tramutola, A. & Trodden, P. (2011). Model predictive control application to spacecraft rendezvous in Mars Sample & Return scenario. In Proc. 4th European Conference for Aerospace Sciences (EUCASS), Saint Petersburg, Russia.

[2] Bemporad, A. & Rocchi, C. (2011). Decentralized linear time-varying model predictive control of a formation of unmanned aerial vehicles. In Proc. 50th IEEE Conf. on Decision and Control and European Control Conf., Orlando, FL, pp. 7488–7493.

[3] Medina, A., Pradalier, C, Paar, G., Merlo, A, Ferraris, S. (2011). A Servicing Rover for Planetary Outpost Assembly. In Proc. 11th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA), April 12-14, Noordwijk, The Netherlands.

[4] Krenn, R., Gibbesch, A (2011). Soft Soil Contact Modeling Technique for Multi-Body System Simulation. In Trends in Computational Contact Mechanics, Vol 58 of Lecture Notes in Applied Computational Mechanics, pp 135-155, Springer, Berlin, Heidelberg.

[5] Binet, G., Krenn, R., Bemorad, A. (2012). Model Predictive Control Applications for Planetary Rovers. In Proc. 11th International Symposium on Artificial Intelligence and Robotics in Space (iSAIRAS), Sept 4-6, Turin, Italy.

[6] Michaud, S., Gibbesch, A., Thuer, T., Krebs A., Lee, C., Despont, B., Schäfer, B., Slade, R. (2008). Development of the ExoMars Chassis and Locomotion Subsystem. In Proc. 9th International Symposium on Artificial Intelligence and Robotics in Space (iSAIRAS), Feb 26-29, Los Angeles, USA.

[7] Bekker, M.G. (1969), Introduction to Terrain-Vehicle Systems, The University of Michigan Press, Ann Arbor.

[8] Janosi, Z., Hanamoto, B. (1961), An Analysis of the Drawbar Pull vs. Slip Relationship for Track Laying Vehicles, Land Locomotion Laboratory, Report RR 47.

[9] Bünte, T., Brembeck, J., Ho, L.M. (2011). Human Machine Interface Concept for Interactive Motion Control of a Highly Maneuverable Robotic Vehicle. In Proc. 4th IEEE Intelligent Vehicles Symposium, June 5-9, Baden-Baden, Germany.