MODEL PREDICTIVE CONTROL APPLICATIONS FOR PLANETARY ROVERS

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

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

ABSTRACT

Model Predictive Control (MPC) is a well-known method for control of processes with low or moderate dynamics as found in power or chemical plants. Within space applications the typical domain of MPC is spacecraft attitude and orbit control. MPC for control of planetary rovers is a quite new technology and was recently investigated in the frame of the RobMPC project, under ESA contract. In this context the Robust-MPC approach was applied to three layers of the rover control hierarchy dealing with medium to high dynamics control tasks: 1) guidance, 2) trajectory control and 3) wheel traction and steering control. The selected reference rover is ESA's four-wheel EGP rover with rear axle steering and a mass of approximately 800 kg.

The MPC control design flow is based on the MPCSofT Toolbox for MATLAB, a novel toolbox developed within the RobMPC project. The MPCSofT toolbox provides an environment for design and simulation 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. As MPC prediction models are easily specified by the user in Embedded MATLAB code, C-code can be automatically generated within the MATLAB/Simulink environment for immediate rapid prototyping.

The highest control level is shared between the nominal path planner (computed offline) and the MPC guidance function. When the rover slips outside the safety corridor around the nominal path, the guidance function continuously builds obstacle-free optimal contingency paths to bring back the vehicle to the nominal path, without the need of stopping the rover to compute a new nominal path. The LTV model included in the MPC optimization engine is used to reconstruct the guidance path from the computed optimal sequence of actions. The MPC trajectory control acts on the velocity vector of the vehicle in order to keep the vehicle within the nominal (guidance) path. This level takes into account the non-holonomic characteristics of the rover and implements a kinematic LTV model of the vehicle. The lowest MPC level is dedicated to traction and steering control. This layer is controlling the steering angle and wheel velocity coordination and replaces typically the Ackermann control. Here, the MPC solution is based on a multi-body system model of the rover including the wheel-soil interaction dynamics. It is implemented as a stepwise LTI class problem with corresponding online linearization of the model.

The paper will introduce the architecture of the entire control hierarchy together with selected details of the MPC specific implementation. The performance and robustness analyses are presented based on results of comprehensive Monte Carlo simulations. A profiling of the code will give an outlook regarding readiness state in terms of controller implementation on space qualified computer hardware.

1. INTRODUCTION

The main advantage of Model Predictive Control (MPC) as a control method for multivariable systems is that constraints are handled in a much more satisfactory way than with other control methods. Unlike most other approaches, MPC is not restricted to Linear Time Invariant (LTI) dynamics. Linear-Time-Varying (LTV) extensions are quite straight-forward and MPC could therefore be a good choice for space control problems which are not amenable to an LTI model such as, for instance, powered landing and launchers applications (decreasing mass) as well as rendezvous and formation flying problems LTV systems.

Within space applications the typical domain of MPC is spacecraft attitude and orbit control, e.g. focused in the ORCSAT project [1], funded by ESA. MPC for control of planetary rovers is a quite new technology and was recently investigated in the frame of the "Robust Model Predictive Control for Space Constrained Systems" (RobMPC) project (see also Section 8 for acknowledgments) Within this activity, in addition to rover locomotion, the application of MPC to collaborative Unmanned Aerial Vehicles has also been treated but is outside the scope of this paper (see e.g. [2]). The scope of RobMPC was to assess the performances of MPC controllers at different levels of rover locomotion control, benchmarking them against reference classic controllers while exploiting the

capabilities inherent to MPC, such as online constrained optimization. In addition, the objective was to validate the robustness of the designed controllers by varying the vehicle and environment parameters outside their nominal values.

In order to assist the design of MPC controllers, the MATLAB/Simulink toolbox MPCSofT was developed within the RobMPC project, which allows the use of LTV state-space models.

2. MODEL PREDICTIVE CONTROL

Model Predictive Control (MPC) is an advanced control technique that explicitly uses a dynamical model of the process to predict its evolution over a given time horizon in the future, and numerical optimization methods to determine the optimal sequence of control inputs that minimizes a given performance index under constraints on input $\mathbf{u}$ and state variables $\mathbf{x}$. The optimization is repeated at each sampling instant, so that the applied optimal input is always based on the latest available measurement data. In this paper we consider *linear time-varying* (LTV) prediction models and constraints as well as quadratic performance functions, which makes MPC implementable by solving a convex quadratic programming (QP) optimization problem.

Let t be the current time expressed in seconds and k be the prediction time expressed in sampling steps, with T_s being the sampling time. The LTV-MPC controller relies on the following rather general linear time-varying prediction model

$$\begin{aligned}\mathbf{x}_{k+1} &= \mathbf{A}\left(t,k,x(t)\right)\mathbf{x}_k + \mathbf{B}\left(t,k,x(t)\right)\mathbf{u}_k + \mathbf{f}\left(t,k,x(t)\right)\\ \mathbf{z}_k &= \mathbf{E}_z\left(t,k,x(t)\right)\mathbf{x}_k + \mathbf{H}_z\left(t,k,x(t)\right)\mathbf{u}_k + \mathbf{P}_z\left(t,k,x(t)\right)\Delta\mathbf{u}_k\\ \mathbf{c}_k &= \mathbf{E}_c\left(t,k,x(t)\right)\mathbf{x}_k + \mathbf{H}_c\left(t,k,x(t)\right)\mathbf{u}_k + \mathbf{P}_c\left(t,k,x(t)\right)\Delta\mathbf{u}_k\end{aligned}$$

where $\mathbf{x}$ is the state vector, $\mathbf{u}$ is the input vector, $\Delta\mathbf{u}_k = \mathbf{u}_k - \mathbf{u}_{k-1}$ is the input increment, $\mathbf{z}$ is the "performance vector" to be optimized, $\mathbf{c}$ is the "constrained vector", and $\mathbf{A}$, $\mathbf{B}$, $\mathbf{f}$, $\mathbf{C}$, $\mathbf{E}_z$, $\mathbf{H}_z$, $\mathbf{P}_z$, $\mathbf{E}_c$, $\mathbf{H}_c$, $\mathbf{P}_c$ are (possibly time-varying and state-dependent) matrices and vectors. The MPC performance index to be minimized at each sampling instant is

$$\rho_1\varepsilon_1^2 + \rho_2\varepsilon_2^2 + \sum_{k=0}^{N(t)-1}\left(\mathbf{z}_k - \mathbf{r}_k\right)^T\left(\mathbf{z}_k - \mathbf{r}_k\right)$$

subject to

$$\begin{aligned}\Delta\mathbf{u}_k &= 0, \quad \forall k = N_u(t), N_u(t+1), \ldots, N(t)\\ \mathbf{c}_k &\le \mathbf{c}_{max}\left(t,k,x(t)\right) + \mathbf{V}_c\left(t,k,x(t)\right)\varepsilon_1, \quad k = 0,\ldots,N(t)-1\\ \mathbf{C}_N\left(t,k,x(t)\right)\mathbf{x}_{N(t)} &\le \mathbf{d}_N\left(t,k,x(t)\right) + \mathbf{V}_N\left(t,k,x(t)\right)\varepsilon_2\end{aligned}$$

where $\mathbf{r}_k$ is the reference signal at time $t + kT_s$ for vector $\mathbf{z}_k$, $N(t)$ is the prediction horizon, $N_u(t)$ is the control horizon, ε_1 and ε_2 are slack variables used to soften the constraints. Constraints can be hardened by zeroing the corresponding entry in vector $\mathbf{V}_c$ and in $\mathbf{V}_N$, where $\mathbf{V}_c \ge 0$ and $\mathbf{V}_N \ge 0$.

The resulting finite-time optimal control problem is mapped into a quadratic programming (QP) problem, in which the optimization variables are $\Delta\mathbf{u}_0, \ldots, \Delta\mathbf{u}_{Nu(t)}$, ε_1 and ε_2. The user can exploit the maximum flexibility offered by the Embedded MATLAB (EML) language to define the LTV prediction model, constraints, cost function, reference signals, and all the parameters appearing in the MPC optimization problem in an EML module. This module is used inside the LTV-MPC Simulink block of the MPCSofT Toolbox, shown in Fig. 1, to construct and solve the problem at each sampling step.

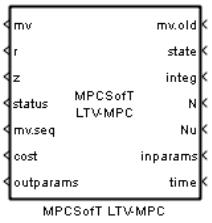

Figure 1. LTV-MPC Simulink block

In fact, the block contains an QP builder function and a QP solver function, both coded in EML code, implementing the LTV-MPC formulation described above. The block is flexible enough to allow an arbitrary number of parameters entering the EML prediction model from the Simulink diagram as real-time varying signals, to vary on-line prediction and control horizons, to limit a priori the maximum number of QP iterations, to include integral action, and to decide the preferred arithmetics (float or double precision). As the block is completely designed based on Simulink blocks and EML functions, C code can be immediately generated for rapid prototyping.

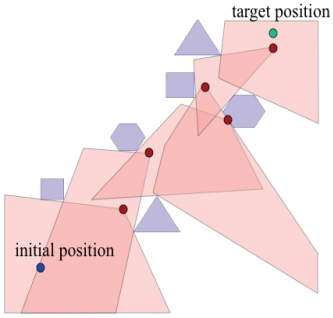

Figure 2. Example of feasible polyhedra generated by the approximation algorithm for obstacle avoidance

The extreme flexibility offered by LTV models is exploited in MPCSofT to impose collision avoidance constraints as linear constraints. An EML function implemented in MPCSofT under-approximates the (possibly nonconvex) feasible space where the vehicle

can navigate by a convex polyhedron. The function is based on a novel and computationally very simple algorithm to determine a (large) polyhedron around the current position of the vehicle that does not contain a set of prescribed polyhedral obstacles, as exemplified in Fig. 2 (see [2] for a detailed description of the algorithm).

3. ROVER REFERENCE APPLICATION

The mission proposed in the rover locomotion (ROL) scenario of RobMPC is a human type mission with secondary scientific targets while fulfilling with the nominal human-related tasks (transport soil, monitoring through images collection, relocating cargo, inspecting the landing site for human arrival and/or the deployed infrastructure, etc.). In this context, the Eurobot Ground Prototype (EGP) (see [3] and [4] and Fig. 3) has been selected as the targeted platform. The main characteristic of the EGP rover are:

- Rear steering wheels;
- Navigation based on IMU sensor;
- Bounding box volume: 2246 x 1580 x 1505 mm (LxWxH);
- Track width: 1330 mm (rear) and 1041 mm (front);
- Wheel stance: 1511 mm;
- Total weight: 880 kg;
- Maximum velocity: 0.35 m/s.

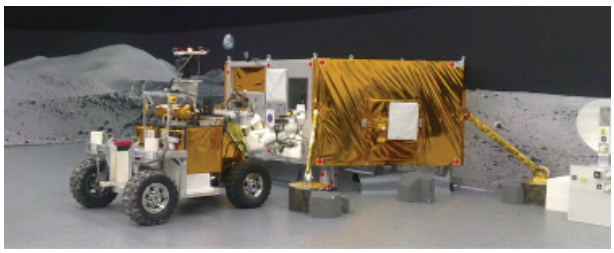

Figure 3. EGP Rover

Concerning the environment in which the rover has been tested, a Martian scenario has been chosen. The soil is supposed to be ideally cohesion-less, which is one of the major differences compared with terrestrial off-road conditions. A playground of 20 x 20 m has been chosen, together with a nominal path to be followed to test the controllers. The nominal path calculated with 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 a number of obstacles.

Within the rover locomotion application, three different MPC controllers have been designed at different hierarchical levels:

- 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.
- 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.
- 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.

This typical rover controllers' hierarchy (as in [5]) is shown in Fig. 4.

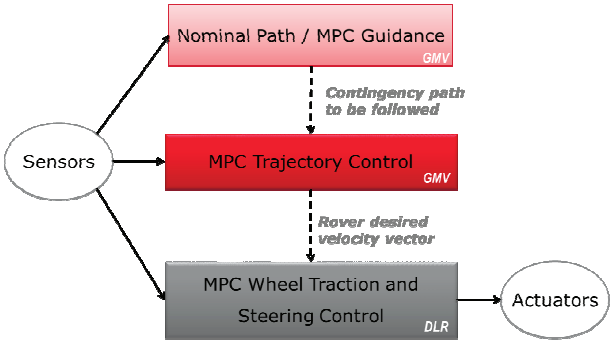

Figure 4. MPC controllers

4. MPC CONTROLLERS DESIGN

This section describes the design of each ROL MPC controller, by addressing the main aspects of an MPC controller. These aspects are:

- The prediction model, which is the core of the MPC prediction engine;
- The reference to be followed by the MPC and the objectives of the MPC optimization;
- The constraints that have been included;
- The guidelines in the choice of the MPC parameters.

4.1. Guidance

The objective of the MPC guidance is to continuously compute contingency paths that bring the rover back into the nominal path. The guidance function is enabled only when the rover finds itself outside the safety path, and is switched off again when the rover is close to the nominal path again (as in Fig. 5). The output of the guidance function is a list of coordinates in terrain reference frame.

Due to the relatively low velocities of the vehicle, a kinematic model has been used to describe the motion of the vehicle, which is modeled as a point. As the model is not linear, this will have to be linearized in order to use LTV MPC control. The model is linearized around the vehicle last measured orientation and velocity. The output of the solution of the MPC gives a

vector which contains the list of actions (linear and angular velocity with respect to reference values) to be applied along the prediction horizon *N*. The linear velocity is assumed to be fixed along the prediction horizon and cannot be controlled; therefore the variable which is of interest is the commanded angular velocity.

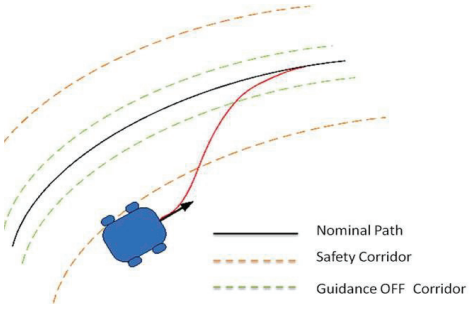

Figure 5. Guidance authority and generated path

The reference to be tracked by the Guidance MPC function is the nominal path, as the objective of this function is to bring back the rover close to the nominal path. At each instant, a list with the position and orientation of the closest nominal path points to the rover are fed to the guidance LTV block. A continuous pre-processing of the nominal path is therefore needed to build a reference vector that contains a list of points which can be targeted by the MPC Guidance function. The spacing of the points of this vector must take into account the velocity and the sample time of the guidance function, as well as the prediction horizon *N*. Weights on the state error and on the input cost will be introduced. As MPCSofT allows varying the parameters of the controller along the prediction horizon, in order to force the guidance to build corrective arcs that inject the rover close to the path in their last points, a bigger cost on the position and heading error is given to the last prediction instant.

The guidance function has to take into account the non-holonomic characteristics of the vehicle in building a contingency path which can be executed by the rover. This is done by setting a maximum absolute value to the angular velocity, which depends on the minimum turning radius of the vehicle and on the nominal linear velocity. The guidance function is called once the rover finds itself outside the safety corridor; therefore the presence of obstacles cannot be excluded. Including the obstacles as constraints raises convexity problems.

The solution which has been adopted in the context of this project is based on the polyhedral approximation function described in Section 2 that is available in MPCSofT. This function creates, for each prediction instant, an allowable obstacle-free space, based on a polyhedron built by estimating the relative position of the obstacles with respect to the vehicle along the prediction horizon. In order to implement this algorithm within the prediction model, a rough prediction of the future states of the rover is needed to calculate the relative position of the obstacles along the prediction horizon *N*. While this is obtained online by simply feeding the previously calculated guidance path, as such a path is not available at the first guidance call; an obstacle-free reactive planner computes this first rough path prediction at the function start-up. In addition to the hard constraint area which takes into account the width of the obstacles and of the rover, an enlarged soft constraint area will be computed around the obstacles it in order to penalize paths which approach obstacles.

The prediction horizon is set depending on the desired length of the guidance path and on the spacing between the guidance points, which is fixed by selecting a minimum and maximum linear velocity. The control horizon N_u has to be set minor or equal to the prediction horizon *N*. Having a N_u equal to *N* would mean that the path can change its heading at each point. However, the sampling time of the trajectory control is sensibly lower than the one of the guidance function. With the consequence that, in fact, the guidance path can be taken as a reference by the trajectory control only for a certain number of guidance path points (depending on the linear velocity and both controllers sampling times), then N_u of the guidance should be chosen accordingly, so that it matches this number of guidance path points. The weights on the error (distance with respect to the path) and on the control input (turns in the path and changes of turns along the path) have been selected in order to bring smoothly the rover back to the nominal path.

4.2. Trajectory Control

The objective of the trajectory control is to keep the rover around a given path, either nominal path or guidance contingency path, with a fixed linear velocity minimizing energy consumption and maneuvers. It controls, therefore, the omega of the vehicle in order to correct the motion of the rover. As the lower layer, the MPC wheel traction and steering controller, controls the motion of a virtual point in front of the vehicle (so called Drawbar) due to the characteristic of the EGP rover (i.e. rear steering wheels), the MPC trajectory control will therefore monitor the position of the Drawbar. In addition, the control of the heading of the vehicle is to be left to the MPC wheel traction and steering controller. Therefore, the orientation which is controlled by the MPC trajectory control is the orientation of the velocity vector of the Drawbar. The output of the MPC trajectory control is the desired velocity vector, expressed in terrain frame coordinates, of the Drawbar point. The orientation of the desired velocity vector is obtained by integrating the desired omega, which is the output of the MPC Trajectory Control block. The module of the desired velocity is set outside the trajectory control module.

Similarly to the guidance function, a kinematic model of the rover is considered to be sufficiently descriptive, due to the low velocities of the system. However, in the trajectory control the linearization approach has been approached in a different way, as when the trajectory control is called it can be assumed that the rover finds itself close to the path to be followed. Therefore it is possible to linearize the rover state around its projected position on the path, which is also called in literature the "reference rover" (see [6]). Note that the trajectory control is transparent to which kind of path it is following, i.e. the trajectory control has no knowledge if it is following the nominal path or the guidance-generated contingency path. Moreover, as the only output of this controller which is effectively used in the angular velocity and the linear velocity is not controlled, this can be excluded by the input vector. As a model of the dynamics of the angular velocity actuators of the rover has been obtained, it is possible to include it in the prediction model of the rover in order to improve the performances of the MPC controller.

With respect to the constraints, the presence of the obstacles must not be taken into account, therefore simplifying considerably the formulation. The minimum turning radius is taken into account in the same way as in the guidance function. In the trajectory control cost function, weights on the position and orientation error have been included. In addition, the cost of omega and the cost of changing omega have been added.

As MPCSofT allows varying the parameters of the controller along the prediction horizon, in order to have a trajectory control which does not stays too far from the path, a bigger cost on the position and heading error is given to the last prediction instant.

4.3. Wheel Traction and Steering Control

The wheel traction and steering control is the interface between the MPC based trajectory control layer and the low level, device specific controllers for individual control of wheel and steering actuators (see Fig. 4). It is a coordinating instance that would replace conventional Ackermann control. In this implementation, which is dedicated for the EGP rover system kinematics and dynamics, the controller computes four individual wheel reference velocities and two individual steering reference angles (six controls **u**) in order to realize the trajectory control requirements in an optimal manner. Since the trajectory control is providing only translational reference velocities the heading angle of the vehicle with respect to the tangent of the desired vehicle path is still freely selectable. Accordingly, the wheel traction and steering control has the option to let the rover drift and slide, which is an important dynamic feature for off-road vehicle locomotion performance.
The controller's prediction model includes a simplified double-track model of the rover. The system has three degrees of freedom (three states **x**), which are the longitudinal and lateral translational vehicle velocities as well as the rotational velocity around the vertical vehicle axis. Accordingly, the system can perform motions on a plane, which may be inclined in order to represent actual terrain shape conditions. The vertical system dynamics is neglected within the double-track model. The wheel-soil interactions in terms of pressure-sinkage relationship and shear stress conditions is described by a non-linear contact dynamics model, which is based on Bekker's semi-empirical terramechanics theory, e.g. [7], and corresponding extensions by Janosi and Hanamoto [8]. In order to be compliant with MPCSofT the non-linear prediction model is numerically linearized around the current state $\mathbf{x}_i$ and control inputs $\mathbf{u}_i$ at each sampling step i during operations. Analytical (symbolical) linearization is not practical for this purpose.

The cost function expresses two control goals. The primary goal is realizing the desired vehicle velocity vector as good as possible under consideration of longitudinal wheel slippage and lateral sliding. Since the primary goal is frequently not achievable under off-road conditions, it is explicitly not implemented as constraint in order to avoid numerical trouble. The secondary goal considers the coordination of wheel velocities and steering angles. Due to the consecutive model linearization around the current conditions $\mathbf{x}_i$ and $\mathbf{u}_i$ during operations the optimization may end up in local optima and get stuck at extreme or exotic configurations. Therefore the secondary goal is maintaining a reasonable configuration of wheel velocities and steering angles. In this application the output of the Ackermann control is taken as reference to be approximated. The importance of the single goals is expressed be weighting factors inside the cost function.

The optimization constraints are the "natural" limitations of the vehicle system, namely the steering angle limits and the load dependent limits of the actuator velocities and accelerations.

The selection of the prediction horizon N is harmonized with the controller sampling step ratio regarding the next higher controller level. Under the assumption that the trajectory controller works approximately by one order of magnitude slower than the wheel traction and steering control, the obvious solution is $N \approx 10$ (see also Section 4.4).

4.4. MPC Controllers Integration

When integrating the MPC controllers, the selection of the controller frequencies has been a demanding task. Typically, choosing the sample time of a controller depends on the dynamics of the system which has to be

controlled and on the dynamics of the sensors and actuators which are available. In addition, the implementation constraints (such as available computational power) have to be taken into account too. In addition, while choosing the correct relative sample times between different MPC controllers which have to be integrated, one has to keep in mind the length of the prediction horizon of each controller in absolute time (which is the sample time multiplied by the length of the prediction horizon *N*).

5. MPC CONTROLLERS EVALUATION

One of the main objectives of RobMPC was to evaluate the robustness of the MPC controllers in space. Therefore, for each MPC controller a robustness validation has been performed. In addition, in order to evaluate the MPC nominal performances, these have been compared against reference controllers. In this context, a number of performance indicators have been designed in order to assess precision and effort of each controller. A high-fidelity functional engineering simulator of the EGP rover has been used to evaluate the MPC performances. First each controller has been evaluated alone, while in a second step the integration of all three MPC controllers has been tested.

5.1. Guidance

As the guidance is triggered when the rover is outside the safety path, different positions have been identified. The results of the MPC guidance function in the nominal tests highlight that the guidance function can successfully compute contingency paths along the whole nominal path. The different position where the MPC guidance function has been tested show that the function can also react to the presence of obstacles by computing collision free paths which bring the rover back into the nominal path safety corridor. However, in order to properly assess the performances of the guidance function, this has to be tested in a scenario where all the other controllers are in the loop, i.e. in an online scenario where the guidance is continuously called as long as the rover is outside the safety path.

In the robustness tests the MPC guidance shows that it successfully builds collision-free contingency paths regardless of the starting orientation of the vehicle. Changing the orientation of the vehicle, the paths can vary depending on the scenario where the guidance function has been called (distance to and shape of the nominal path, obstacles to be avoided). When the starting orientation of the path does not permit, due to the minimum turning radius restrictions or to the position of the obstacles, to create a contingency path which brings the rover into the safety corridor around the nominal path, then the MPC guidance will compute a path which better puts the rover for a following approach to the nominal path (as in Fig. 6).

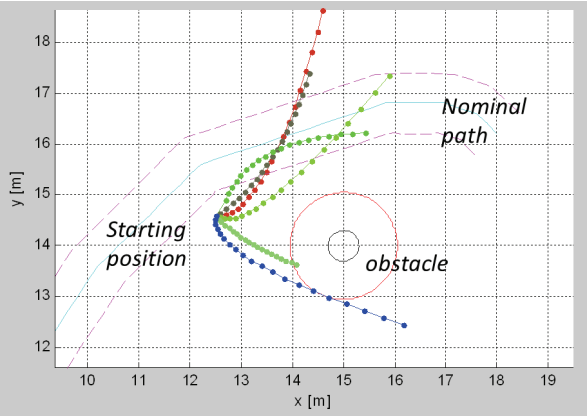

Figure 6. Guidance collision free paths with different starting orientations

5.2. Trajectory Control

The scenario where the trajectory control tests have been performed is of a nominal path to be followed which contains a sharp turn and at least two changes of directions. A pure pursuit algorithm has been used in the PID reference controller (see [9]). In the tests, while both controllers achieve to keep the rover around the nominal path as desired, the MPC shows significantly better results in the control error indicators. For instance MPC showed much better performances in the Integral of Square Error (ISE) indicator, having a result an order of magnitude better than the reference controller. It can be also noted that this difference in performances is mostly caused by the behavior of the controllers when a sharp turn is requires in the nominal path: the MPC can predict and better adjust to it, while the pure pursuit can only react (as highlighted by Fig. 7).

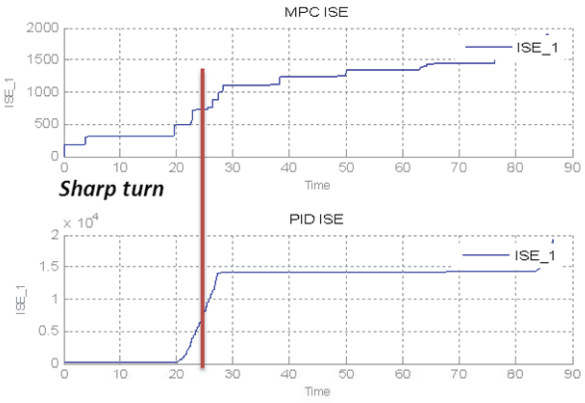

Figure 7. MPC vs PID trajectory controllers ISE, sharp turn is highlighted in red

In addition, the MPC controller also achieves in reaching a much higher precision while also optimizing the control cost (as per Integral of Absolute Derivative of control signal **u**), which in this case is evaluated as the commanded angular speeds of the vehicle. This is not the case for the PID controller, which can get similar performances only with a greater cost.

In the robustness tests, the vehicle parameters (such as rover mass, sensors and actuators precision) and the nominal velocity of the rover have been varied. Robustness has been achieved as in the Monte Carlo simulations campaign the rover has been always kept within the safety corridor around the nominal path by the MPC trajectory controller.

5.3. Wheel Traction and Steering Control

The evaluation of the traction and steering control performance was done using a ramp like terrain with variable inclination and a U-shaped path to be followed by the rover at constant velocity (see Fig. 8).

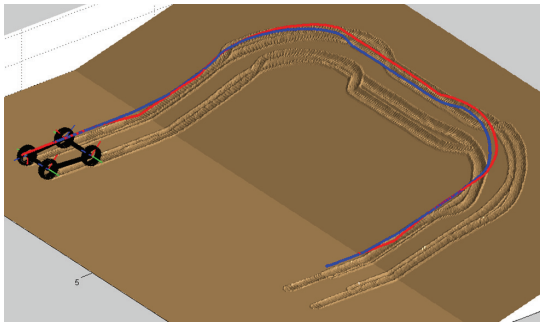

Figure 8. Test scenario for controller evaluation

Under nominal conditions the performance of the MPC-based controller was evaluated versus conventional Ackermann control, which is subject to be replaced. The control performance was expressed by numerous indicators for control accuracy, control effort and control smoothness. Herein, the integral of the square of the vehicle velocity error (ISE) is the most important one. Using these indicators it could be demonstrated that MPC solutions will significantly improve the control performance (see Fig. 9), even if Ackermann control is additionally extended by closed-loop PID style velocity and heading angle controllers. The fact that MPC coordinates all (six) actuators individually is obviously the key feature for improving the control performance compared to Ackermann control, which applies only two controls, namely track radius and track velocity.

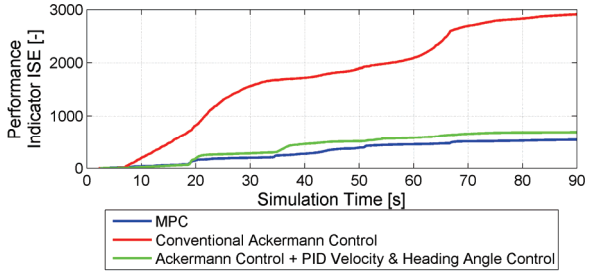

Figure 9. ISE function versus operation time

The robustness of the MPC based controller in terms of uncertainties regarding vehicle parameters (e.g. mass, assembly errors, actuator performance variability) and environment parameters (e.g. terrain shape, soil parameters) but also in terms of controller parameters (e.g. sampling time, prediction horizon) was successfully proven by extensive Monte Carlo analyses. In Fig. 10 the results for the most relevant parameters of each group are presented.

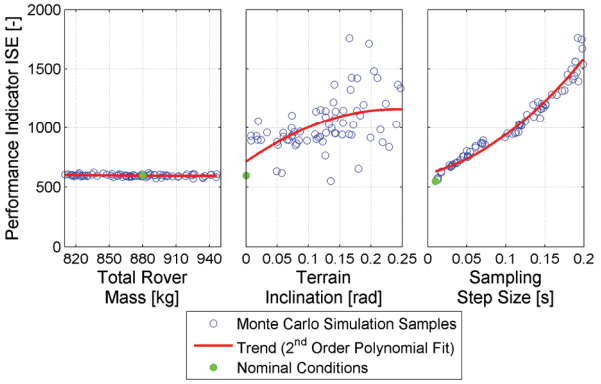

Figure 10. ISE versus most relevant parameters

Accordingly, one can conclude the following: Vehicle parameter uncertainties are almost negligible for controller tuning. In terms of environment parameters the terrain inclination is an important issue. However, it is not an explicit controller parameter and following not subject of controller tuning. A very clear trend can be identified regarding the controller sampling time, which suggests fast sampling of the controller and brings the focus on issues like computational performance of space-qualified computer hardware and optimal sampling time distribution inside the entire controller hierarchy (see also Sections 4.4 and 5.4).

5.4. Integrated Controllers

The integration of all three MPC controllers has been 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 into obstacles but with the other two MPC controllers in the loop. This has been tested against a scenario where no guidance function has been implemented and the nominal path is kept as the only reference to be tracked. The obtained results, tested in several different positions in a representative scenario, showed that the guidance function successfully creates obstacle-free paths which are followed by the other controllers and bring the rover back to the nominal path. A clear advantage is that the MPC guidance avoids stopping the rover in order to re-compute a nominal path, a typically time and energy consuming task which often requires interaction with the operator. The guidance function therefore enhances the autonomy of the system. In addition, it has been shown that guidance generated contingency paths allow an effective and smooth motion of the vehicle towards the nominal path.

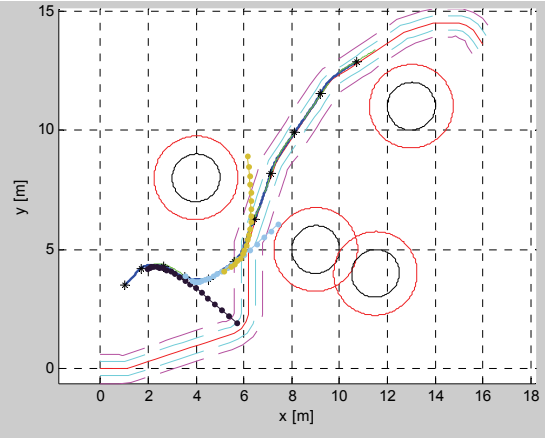

Figure 11. All MPC controllers working together

The configuration with the three MPC controllers in the loop has been tested in different scenarios (e.g. Fig. 11) and, also, its robustness has been positively validated against the variation of the vehicle and scenario parameters, such as sensors and actuators errors, terrain inclination and starting vehicle position.

6. HARDWARE IMPLEMENTATION ASPECTS

The evaluations work was performed using a functional engineering simulator (see Section 5), which implicitly solved two types of challenges to be considered in case of hardware-implementation: Measurability or observability of required sensor feedback and real-time performance of controllers. Regarding sensor data rover applications can benefit from developments made for autonomous navigation of ground vehicles. However, in terms of computational power the MPC implementation has still hard constraints form available space qualified computer hardware. With the proposed sampling step relationship of 1/10/100 Hz of the three MPC levels and according to preliminary code profiling work one has to state that so far only guidance and trajectory control are implementable on standard space hardware. In case of traction and steering control FPGA implementation seems to be the most realistic solution. Alternatively, the sampling rate could be decreased, which was proven to be possible while maintaining an acceptable control performance.

7. CONCLUSIONS

In the context of the project RobMPC (see also Section 8 for acknowledgements) it could convincingly be demonstrated that MPC is a promising solution for all layers of the rover locomotion control hierarchy: Guidance, trajectory control and wheel traction and steering control. The three developed MPC solutions were validated versus corresponding standard solutions and their respective robustness was proven by comprehensive Monte Carlo simulations. Beside the individual performance of the MPC layers their mutual compatibility was ensured as well. The features of MPCSofT support the specific needs of the particular controller architectures like LTI and LTV MPC as well as the user-friendly definition of cost functions and constraints. Since the controllers are implemented using Embedded MATLAB code they are already prepared for running on a rover's onboard computer under real-time conditions. Based on the promising results of RobMPC the project partners have proposed to include the MPC topic in the ESA technology harmonization process for further support of development and hardware implementation.

8. ACKNOWLEDGEMENT

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

9. REFERENCES

1. M. Saponara, V. Barrena, A. Bemporad, E.N. Hartley, J. Maciejowski, A. Richards, A. Tramutola, and P. Trodden, Model predictive control application to spacecraft rendezvous in Mars Sample & Return scenario," *Proc. 4th European Conference for Aerospace Sciences (EUCASS)*, Saint Petersburg, Russia, 2011.
2. A. Bemporad, C. Rocchi, Decentralized linear time-varying model predictive control of a formation of unmanned aerial vehicles, *Proc. 50th IEEE Conf. on Decision and Control and European Control Conf.*, Orlando, FL, 2011, pp. 7488–7493.
3. A.Medina, L.Mollinedo, F.Gandía, C.Pradalier, G.Paar, E.Pensavalle, F.Didot, P.Schoonejans (2012). Integrated autonomous navigation in a large planetary exploration and service rover. *iSairas 2012*.
4. Medina, A., Pradalier, C; Paar, G.; Merlo, A; Ferraris, S. (2011). A Servicing Rover for Planetary Outpost Assembly. *Astra 2011*.
5. Raffo G. V., Gomes G.K., Normey-Rico J.E., Kelber C.R., Becker L.B. (2009) A Predictive Controller for Autonomous Vehicle Path Tracking, *Intelligent Transportation systems vol 10 Issue 1*.
6. Kühne F., Gomes da Silva J.M., Lages W.F. (2005) Mobile Robot Trajectory Tracking Using Model Predictive Control, *VII SBAI / II IEEE LARS.*
7. Bekker, M.G. (1969), Introduction to Terrain-Vehicle Systems*, The University of Michigan Press, Ann Arbor, USA*.
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. Coulter, C. (1992), Implementation of the Pure Pursuit Path Tracking Algorithm. *The Robotics Institute Carnegie Mellon University.*