Science Arts & Métiers (SAM)

is an open access repository that collects the work of Arts et Métiers Institute of Technology researchers and makes it freely available over the web where possible.

This is an author-deposited version published in: https://sam.ensam.eu
Handle ID: .http://hdl.handle.net/10985/16315

To cite this version :

Carolina RENGIFO, Jean-Rémy CHARDONNET, Hakim MOHELLEBI, Damien PAILLOT, Andras KEMENY - Feasibility Analysis For Constrained Model Predictive Control Based Motion Cueing Algorithm - In: 2019 International Conference on Robotics and Automation (ICRA), Canada, 2019-05-20 - IEEE International Conference on Robotics and Automation (ICRA) - 2019

Any correspondence concerning this service should be sent to the repository
Administrator : archiveouverte@ensam.eu

Feasibility Analysis For Constrained Model Predictive Control Based Motion Cueing Algorithm

Carolina Rengifo[1,2], Jean-Rémy Chardonnet[2], Hakim Mohellebi[1], Damien Paillot[3] and Andras Kemeny[1,2]

Abstract—**This paper deals with motion control for an 8-degree-of-freedom (DOF) high performance driving simulator. We formulate a constrained optimal control that defines the dynamical behavior of the system. Furthermore, the paper brings together various methodologies for addressing feasibility issues arising in implicit model predictive control-based motion cueing algorithms.**

The implementation of different techniques is described and discussed subsequently. Several simulations are carried out in the simulator platform. It is observed that the only technique that can provide ensured closed-loop stability by assuring feasibility over all prediction horizons is a braking law that basically saturates the control inputs in the constrained form.

I. INTRODUCTION

The actuator capabilities and the workspace limits of driving simulators do not allow motion signals to be sent directly from the driver to the simulator platform. Therefore, the so-called Motion Cueing Algorithms (MCA) were created. Their aim is to transform all trajectories generated by a dynamic virtual model into the desired motion cues within the physical limits by providing the most realistic motion cues.

Model Predictive Control (MPC) is one of few suitable methods to handle this issue. The main advantage of the MPC compared to previously used algorithms (classical, adaptive and optimal) is its ability to handle input and state constraints. Unlike optimal control techniques such as LQR (Linear Quadratic Regulator), an MPC controller does not guarantee closed-loop stability. The optimization problem must guarantee a solution along the prediction horizon, otherwise the applied control law could make the system states become unstable, causing damage to the simulator's structure or even worse, to the driver. When there are only restrictions in the input, a simple saturation can reassure the solution, but when there are state constraints, the task becomes more difficult since there is an inconsistency between the index performance and the constraints.

One of the first techniques that guarantees closed-loop stability if feasibility is also guaranteed is to complement the cost function with a terminal state constraint equal to zero [1]. Another approach is to introduce a terminal penalty equal to the infinite horizon cost [2]. This condition can refer to the solution of the Riccati difference equation as the terminal penalty matrix in the cost function.

Regarding MCA, the MPC was first applied in [3], in which the explicit offline procedure proposed in [4] is used to find the invariant set of initial states from which the system reaches an equilibrium point within a finite time. This problem creates a subset of points defined for a bang-bang control condition where the closed-loop system can achieve states stability within their boundaries. In [5] an explicit strategy for a 1 DOF optimization is implemented, extended in [6] to a 2 DOF optimization problem using an implicit strategy, in which a new stability condition is created based on a braking law. In [7] a motion blocking strategy is implemented by using different sampling frequencies in the prediction. Other authors who use implicit MPC do not make evident the conditions for which the problem is feasible [8], [9] or simply consider states weighting as an alternative to the existence of a control law [10].

It was shown that for processes with fast dynamics it is preferable to implement explicit control strategies in which the control law is found offline and applied online according to the system initial states at each time-step [9]. But this technique with LQ invariant sets is undesirable for the trajectory tracking problem using a 2 DOF optimization, as the computation of the admissible set may not converge [5]. Therefore, in this paper, we will focus on different conditions applied to create a standard implicit MPC-based MCA that can handle constraints, maximize the platform's working space and stabilize the closed-loop system by finding a feasible solution for the quadratic optimization problem (QP). As far as we know, this is the first work that shows a comparison between the available strategies.

This article is organized as follows: in section II we present the mathematical model based on the platform and the human perception motion. Section III offers a design to the MPC-based MCA problem. Section IV provides some appreciations of different strategies implemented in the MCA that are dealing with feasibility issues and in Section V we present all simulations results. The final section concludes the paper.

II. MATHEMATICAL MODEL

In this study, Renault's ULTIMATE 8-DOF motion simulator is considered, allowing three translations and rotations along the x, y, and z axes from a hexapod platform and two supplementary X and Y rails. The simulator is divided into four subsystems, two independent DOFs corresponding to the vertical axis and yaw and two coupled subsystems, longitudinal/pitch and lateral/roll.

---

[1] Renault, Virtual Reality and Immersive Simulation Center, 78288 Guyancourt, France {`carolina.c.rengifo`, `hakim.mohellebi`, `andras.kemeny`}`@renault.com`

[2] LISPEN EA7515, Arts et Métiers, HESAM, UBFC, Institut Image, 71100 Chalon-sur-Saône, France. `jean-remy.chardonnet@ensam.eu`

[3] Université de Bourgogne, LISPEN EA7515, Arts et Métiers, HESAM, UBFC, Institut Image, 71100 Chalon-sur-Saône, France `damien.paillot@u-bourgogne.fr`

A. Platform dynamics

The driving simulator in this study is considered as an ideal one, i.e., the dynamics of the platform is made by the manufacturer and is perfect, therefore, the system could be represented by a unit gain. However, in order to introduce directly the platform position, velocity and acceleration, the model for each linear x, y, z and rotational accelerations is a double integrator with direct feed-through.

$$
\begin{aligned}
x_{sim}(k+1) &= \overbrace{\begin{bmatrix} 1 & t_s & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & t_s \\ 0 & 0 & 0 & 1 \end{bmatrix}}^{A_{sim}} \overbrace{\begin{bmatrix} p(k) \\ v(k) \\ \theta(k) \\ \omega(k) \end{bmatrix}}^{x_{sim}} + \overbrace{\begin{bmatrix} \frac{t_s^2}{2} & 0 \\ t_s & 0 \\ 0 & \frac{t_s^2}{2} \\ 0 & t_s \end{bmatrix}}^{B_{sim}} \overbrace{\begin{bmatrix} u_{lin} \\ u_{rot} \end{bmatrix}}^{u} \\
y_{sim} &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_{lin} \\ u_{rot} \end{bmatrix}
\end{aligned}
\tag{1}
$$

We opted for this representation rather than a simple unit gain in order to integrate the platform constraints into the optimization problem. The states are the position p, speed v, angle θ and angular speed ω. The outputs are the linear u_{lin} and rotational u_{rot} accelerations. Table I exposes the performance of the dynamic platform along all degrees of freedom.

TABLE I: Performance of Renault's ULTIMATE simulator

Actuator/Rail limits	**Displacement**	**Velocity**	**Acceleration**
Longitudinal/x	$0.28m$	$0.7m/s$	$7.5m/s^2$
Lateral/y	$0.26m$	$0.7m/s$	$7.5m/s^2$
Vertical/z	$0.20m$	$0.4m/s$	$5.0m/s^2$
Roll	$15°$	$40°/s$	$300°/s^2$
Pitch	$15°$	$40°/s$	$300°/s^2$
Yaw	$15°$	$60°/s^2$	$600°/s^2$
X rail	$2.6m$	$2m/s$	$5m/s^2$
Y rail	$2.6m$	$3m/s$	$5m/s^2$

B. Human perception motion

In this part we use human perception to drive the motion simulation, i.e., instead of taking the accelerations directly from the virtual dynamic model, we will track the path of the motion accelerations perceived in the simulator using a human perception model.

The main motion sensors in humans are found in the vestibular system which comprises two components. The first is the semi-circular canals used to indicate rotational accelerations and the second is the otoliths organs which sense linear accelerations. The total perceived acceleration (the rotational acceleration and the linear acceleration) leads to a force, called the specific force.

For the semicircular canals a transfer function that links the angular velocity perceived by the driver $\hat{\omega}$ and the real angular velocity ω is used as follows:

$$
\frac{\hat{\omega}}{\omega} = \frac{G_{\text{csc}} \tau_L \tau_a s^2 (1 + \tau_l s)}{(1 + \tau_a s)(1 + \tau_L s)(1 + \tau_s s)} \tag{2}
$$

This model contains different time constants that are mostly based on subjective responses: the long time constant τ_L, the short time constant τ_s, an adaptation operator τ_a and the lead term τ_l to avoid vibration effects. We use the values and model described in more detail in [11]. This transfer function is implemented as a filter for the three rotations angles along the x, y, z axes; roll, pitch and yaw respectively. The result is added to the tilt angles from the specific forces and then saturated to avoid exceeding the limits in Table I.

For the otoliths model, the input is the specific force f (head linear acceleration) and the output is the sensed specific force $\hat{f}$. The transfer function that best represents the relationship between these two stimuli is:

$$
\frac{\hat{f}}{f} = \frac{G_{oto}(\tau_{aoto} s + 1)}{(\tau_{Loto} s + 1)(\tau_{soto} s + 1)} \tag{3}
$$

The parameters in (3) are taken from the same reference as the one used for the semicircular canals [11].

The otoliths perceive both linear acceleration and head inclination, as a linear acceleration. This discrepancy allows us to restore an illusory sensation of linear accelerations by tilting the platform as shown in Fig. 1. A threshold for motion perception must be integrated so that the driver does not perceive the vestibular cues.

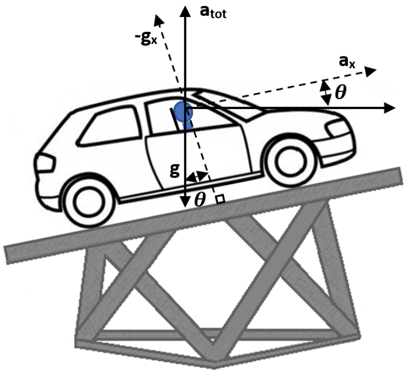

Fig. 1: Tilt platform along the longitudinal acceleration

In that sense, the linear specific force along the three axes is perceived by the otoliths as

$$
f_x = a_x + g sin(\theta) \tag{4}
$$

$$
f_y = a_y - g sin(\phi) cos(\theta) \tag{5}
$$

$$
f_z = a_z - g cos(\theta) cos(\phi) \tag{6}
$$

where a_x, a_y, a_z, represents the translation acceleration vector components along the x, y and z axes, ϕ and θ are respectively, the rotation angle around the y and x axes and g represents the gravity.

Now, we will only take the x axis as an example since the others are done in the same way. Assuming small angles, the Laplace form for the specific force is

$$
f_x(s) = u_{lin}(s) + \frac{g}{s^2} u_{rot}(s) \tag{7}
$$

Replacing (7) into (3) we obtain

$$
\hat{f}_x = \frac{G_{oto}(\tau_{aoto} s + 1)}{(\tau_{Loto} s + 1)(\tau_{soto} s + 1)} \left[u_{lin}(s) + \frac{g}{s^2} u_{rot}(s) \right] \tag{8}
$$

and the system space states form

$$\dot{x_{oto}} = \overbrace{\begin{bmatrix} -T_{3oto} & 1 & 0 & 0 \\ -T_{4oto} & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}}^{A_{oto}} x_{oto} + \overbrace{\begin{bmatrix} T_{1oto} & 0 \\ T_{2oto} & 0 \\ 0 & gT_{1oto} \\ 0 & gT_{2oto} \end{bmatrix}}^{B_{oto}} \begin{bmatrix} u_{lin} \\ u_{rot} \end{bmatrix}$$

$$y_{oto} = \overbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}}^{C_{oto}} x_{oto} \tag{9}$$

where

$$T_{1oto} = \frac{G_{oto}\tau_{aoto}}{\tau_{Loto}\tau_{soto}}, \qquad T_{2oto} = \frac{G_{oto}}{\tau_{Loto}\tau_{soto}},$$
$$T_{3oto} = \frac{\tau_{Loto} + \tau_{soto}}{\tau_{Loto}\tau_{soto}}, \qquad T_{4oto} = \frac{1}{\tau_{Loto}\tau_{soto}}$$

The system (9) is augmented with the platform dynamics (1) in order to include each simulator state and thus add the physical constraints in each period of time. With this modification, we get a unique system for the longitudinal/pitch and the lateral/roll DOFs in which the output $\hat{f}$ is obtained from the inputs, u_{lin} and u_{rot}.

$$\begin{aligned} \dot{X_{force}} &= \overbrace{\begin{bmatrix} A_{oto} & 0 \\ 0 & A_{sim} \end{bmatrix}}^{A_{force}} \overbrace{\begin{bmatrix} x_{oto} \\ x_{sim} \end{bmatrix}}^{X_{force}} + \overbrace{\begin{bmatrix} B_{oto} \\ B_{sim} \end{bmatrix}}^{B_{force}} \overbrace{\begin{bmatrix} u_{lin} \\ u_{rot} \end{bmatrix}}^{u} \\ Y_{force} &= \overbrace{\begin{bmatrix} C_{oto} & 0 \end{bmatrix}}^{C_{force}} \begin{bmatrix} x_{oto} \\ x_{sim} \end{bmatrix} + \overbrace{\begin{bmatrix} 0 & 0 \end{bmatrix}}^{D_{force}} \begin{bmatrix} u_{lin} \\ u_{rot} \end{bmatrix} \end{aligned} \tag{10}$$

III. CONTROL DESIGN

The system (10) is represented as a linear time-invariant (LTI) discrete-time system and is augmented with an embedded integrator to give offset free tracking (see Appendix).

$$\begin{aligned} x_{k+1} &= Ax_k + B\Delta u_k \\ y_k &= Cx_k + D\Delta u_k \end{aligned} \tag{11}$$

with state vector $x_k \in \mathbb{R}^s$, control $\Delta u_k \in \mathbb{R}^l$, output $y_k \in \mathbb{R}^t$ and matrices (A, B) of compatible dimensions.

The vectors of the predicted output Y, the states X and the future control ΔU are defined as follows:

$$\begin{aligned} X &= \begin{bmatrix} x(k+1) & x(k+2) & \cdots & x(k+N_p) \end{bmatrix}^T \\ Y &= \begin{bmatrix} y(k+1) & y(k+2) & \cdots & y(k+N_p) \end{bmatrix}^T \\ \Delta U &= \begin{bmatrix} \Delta u(k) & \Delta u(k+1) & \cdots & \Delta u(k+N_u-1) \end{bmatrix}^T \end{aligned} \tag{12}$$

where N_p denotes the prediction horizon $(1 \leq N_p)$, N_u denotes the control horizon $(0 < N_u \leq N_p)$.

In terms of current states and future control increments, the predictions take the form:

$$\begin{aligned} Y &= Fx_k + G\Delta U \\ X &= F_x x_k + G_x \Delta U \end{aligned} \tag{13}$$

The matrices F, F_x, G, G_x are found recursively and depend on the state space matrices. Then, the tracking objective function implemented as a 2 DOF optimization to improve motion fidelity is defined in the vector notation as

$$J = (Y - R_s)^T Q_\delta (Y - R_s) + \Delta U^T Q_\lambda \Delta U + X^T Q_q X \tag{14}$$

subject to:

$$\begin{aligned} x_{min} &\leq X \leq x_{max} \\ y_{min} &\leq Y \leq y_{max} \\ \Delta u_{min} &\leq \Delta U \leq \Delta u_{max} \end{aligned} \tag{15}$$

where $Q_\delta > 0$, $Q_\lambda > 0$ and $Q_q > 0$ are symmetric weighting matrices of compatible dimensions for the tracking error (the specific force deviation perceived during the simulation), the control rate and the states respectively. With a given set-point signal $r(k)$ at sample time k, we get the reference trajectory R_s. Since in most cases it is difficult to have the future reference available as it depends on the driver's behavior, R_s remains constant over the prediction horizon N_p.

$$R_s^T = r(k) \times [1, \cdots, 1]_{1 \times N_p} \tag{16}$$

Replacing from (14) the predicted output Y, the predicted states X and removing all the terms that do not depend on the decision variables vector ΔU [12], the cost function obtained has the form

$$\begin{aligned} J \;\; = \;\; & \Delta U^T \left(G^T Q_\delta G + Q_\lambda + G_x{}^T Q_q G_x \right) \Delta U \\ & + 2 \left((Fx(k) - R_s)^T Q_\delta G + (F_x x(k))^T Q_q G_x \right) \Delta U \end{aligned}$$

The QP problem with constraints remains:

$$\begin{aligned} \text{minimize} \quad & J = \frac{1}{2} \Delta U^T H \Delta U + f^T \Delta U \\ \text{subject to} \quad & A_c \Delta U \leq b \end{aligned} \tag{17}$$

where,

$$\Delta U \in \mathbb{R}^n, \quad f \in \mathbb{R}^n, \quad b \in \mathbb{R}^m, \quad A_c \in \mathbb{R}^{m \times n}$$

The H matrix is symmetric positive definite $n \times n$. The matrix equation $A_c \Delta U \leq b$ contains all the linear inequality constraints (15), therefore the feasible set is polyhedral.

We consider two different constraints types: hard and soft constraints. The first one cannot under any condition be violated and they include all platform physical constraints (15). The second one should be satisfied if possible and are relaxed when the hard constraints are conflicting; the weighting matrices are part of this group and these are presented in the cost function to penalize their violation. In our problem the states and input constraints are referred as both hard and soft constraints.

In the MPC, an open-loop control action sequence ΔU is obtained by solving, at each sampling time, a cost function (17) over a finite horizon N_p. Only the first optimal control in the sequence is applied to the plant to ensure a feedback loop while the remaining optimal inputs are discarded. The next time step, a new optimal control sequence is solved with a different initial state, leading to a receding horizon control [13]. An active set method implemented in the qpOASES tool [14] has been chosen to deal with the optimization.

A. *Nonuniform prediction window*

When the reference signal is a positive acceleration and the platform is close to its limits, the platform must slow down with an acceleration below the human motion threshold to prevent the driver from perceiving false cues. This technique is called washout. Based on the literature we use $-0.2m/s^2$ as the acceleration limit in the washout phase [15].

According to previous MPC-based MCAs [16] and taking into account the limits of the platform shown in Table I, an N_p between 7-14 seconds can be enough for the platform with our technical conditions to stop moving before reaching its limits in position. Considering this interval and a control frequency of 100 Hz (required for a real-time implementation), we get an optimization problem that cannot been solved in real time due to high computational cost.

For this purpose, we use a nonuniform sampling frequency along the prediction window. The method consists in applying two different frequencies when dicretizing the system and the constraints (11). The first sampling period t_{s1} is the same as the optimization step (0.01s) and will be fixed in the first 10 steps. For the second t_{s2} a simulation is performed: we fix the prediction time to 7.1s, and vary the prediction horizon N_p and t_{s2} in order to achieve the required prediction time. Table II summarizes the simulation conditions.

TABLE II: Different sampling frequencies for a same prediction time

t_{s1}***(size)**	t_{s2}***(size)**	N_p **size**	**Prediction time (s)**
0.01*(10)	0.5*(140)	150	7.1
0.01*(10)	0.1*(70)	80	7.1
0.01*(10)	0.15*(47)	57	7.1
0.01*(10)	0.2*(35)	45	7.1

Figure 2 shows the closed-loop responses using different frequencies in the nonuniform prediction horizon. The input corresponds to scenario 3 defined in section V. We see that the smaller t_{s2}, the greater the reference tracking, however it generates a higher computational cost since the size of N_p is bigger. We choose then 0.1 for t_{s2}, as it is small enough to capture adequately the dynamics of the process without any oscillations, allowing real-time implementation and good trajectory tracking. We also note from Fig. 2 that the reference is not fully followed, as there are strong restrictions in the states.

B. *Algorithm tuning*

The states and human motion thresholds are too restrictive and have influence on the region for which the problem can find a valid solution ΔU. Therefore we use tuning parameters that will define the control law and therefore the behavior of the closed-loop system.

Tuning consists in choosing the weighting matrices Q_δ, Q_λ, Q_q and the horizons N_p, N_u from the target function (14). Unfortunately, decision making is intuitive and based on trial-error experiences. In this study, all parameters will remain the same for Section IV.

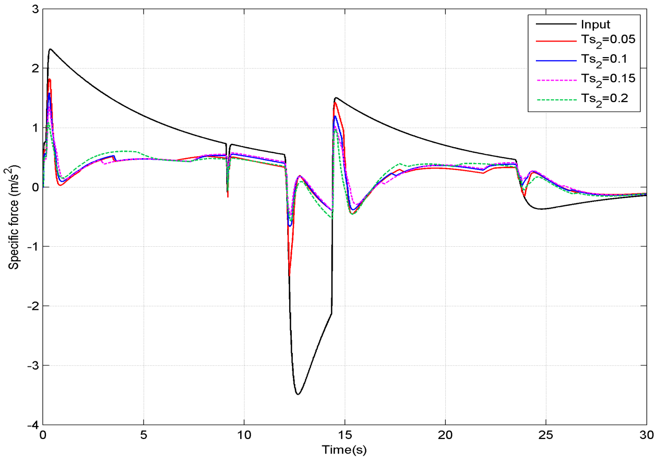

Fig. 2: Closed-loop responses using different frequencies in the nonuniform prediction horizon

IV. COMPARATIVE STRATEGIES

This section aims at explaining and showing some of the strategies applied in the MPC-based MCA to enforce feasibility and possibly ensure stability, since the Lyapounov stability proof applies in the presence of constraints if the solution is feasible over the whole prediction horizon. Normally two techniques are used, the terminal constraints and the terminal penalties in the cost function. We consider for all techniques the same tuning parameters mentioned in the previous section. The terminal penalty may vary depending on the implemented technique.

A. *Terminal zero state constraints*

This is the first method to guarantee stability to nonlinear MPC problems. The idea is to force all platform physical limits to return to the origin (zero in this case). The condition is applied directly in the optimization problem (17) in the form of hard constraints:

$$x(t+N_p)=0 \tag{18}$$

B. *Invariant set*

The equality zero terminal constraint can be relaxed by using a potential constraint set for the constrained LQ optimal control problem. There is a possible set of initial states in which the constraints are respected along the system trajectory. It is set around the origin and constrains the terminal state to stay in the set. The set is polyhedral, i.e., defined by linear inequalities, thus it can easily be put into the QP problem.

This region X_F aims at changing the states constraints at every sample time by following a washout technique condition [3]:

$$X_f=\begin{cases} p_k \le p_{max}-\frac{v_k^2}{2a_s}, & v_k \ge 0 \\ p_k \le \frac{v_k^2}{2a_s}-p_{max}, & v_k < 0 \end{cases} \tag{19}$$

where p_k and v_k represent the position and velocity at sample time k and a_s is the acceleration threshold. The same analogy is implemented for the angle constraints.

C. Braking law

Fang et al. proposed a different stability condition [16]. Their approach can verify a solution along the prediction horizon using a braking law once the platform approaches its limits. This law can be compared to an adaptive filter that modifies the acceleration limits at each sample time depending on the current position p_k/θ_k, the current velocity $v_k/\dot{\theta}_k$ and their respective limits. This law allows the platform to return to its neutral position with a certain threshold while respecting the limits in position, velocity and acceleration. This condition is summarized as:

$$p_{min} \leq p_k + c_v T v_k + c_a \frac{T^2 ulin_k}{2} \leq p_{max} \tag{20}$$

$$\theta_{min} \leq \theta_k + c_v T \omega_k + c_a \frac{T^2 urot_k}{2} \leq \theta_{max} \tag{21}$$

where c_v, c_a and T are tuning parameters that prevent the platform from exceeding its limits. We chose the parameters, so that there would always be a valid solution along the prediction horizon, these are: $c_v = 4$, $c_a = 2$ and $T = 1.2$. The difference between this technique and the invariant set one is that here the constraints are applied more strictly to the control inputs.

D. Terminal weighting matrix based on the LQR theory

Consider the following infinite unconstrained optimal control problem for the system (A, B) to be stabilizable

$$J_k = \frac{1}{2}\sum_{j=1}^{N_p-1} x_k^T Q x_k + \Delta u_k R \Delta u_k + \frac{1}{2} x_{Np}^T P x_{Np} \tag{22}$$

where P is the solution for an infinite horizon and the unique positive solution of the discrete algebraic Riccati equation

$$P = A^T P - K^T (R + B^T P B) K + Q \tag{23}$$

and $K = (R + B^T P B)^{-1} B^T P A$ is the corresponding gain.

Using P as the terminal state weighting matrix for the finite time predicted horizon implies that the cost functions is Lyapunov and then guarantees nominal stability in closed loop. Here, the receding horizon controller with a large enough control and prediction horizon behaves like an LQ optimal control [13].

In the presence of constraints, the controller $\Delta U = Kx$ does not necessarily satisfy the limits of the platform. In that case, it is necessary to change the choice of Q and R weighting matrices as these are the ones that will determine the performance of the system in closed loop.

V. SIMULATION RESULTS

The SCANeR studio driving simulation software is applied to generate different test scenarios and reference signals like longitudinal and rotational accelerations. These signals are sent from a virtual vehicle model to be processed by the MCA and then sent to the simulation platform.

Three different scenarios that last 50 seconds each are compared to evaluate the feasibility of the optimal solution for the different techniques of Section IV. All scenarios require both longitudinal motion with braking/acceleration and lateral motion with steering vehicle control.

The first scenario Sce_1 is a urban simulation consisting in one intersection in which the vehicle must stop and then turn left. The second scenario Sce_2 is a highway simulation: the vehicle goes at a constant speed following a car, passing it and cutting into the right lane. The last scenario Sce_3 is a slalom type scenario with a constant acceleration of $2m/s^2$ during 10 seconds then a stop until null acceleration.

Table III shows the number of infeasibilities for all strategies of Section IV and for each scenario described above.

TABLE III: Number of infeasibilities per strategy in different scenarios

Strategies	Infeasibilities number		
	$Sce1$	$Sce2$	$Sce3$
Zero terminal (S_1)	0	8	1
Invariant condition	0	112	4
Invariant with P matrix (S_2)	0	17	3
Braking law (S_3)	0	0	0
Normal constraints	12	1212	345
Normal with P matrix (S_4)	0	18	2

Restricting the states to be zero at the end of the horizon (S_1) results in very bad tracking as we can see in Figs. 3a, 3b and 3c since the limits on human perception prohibit high variations in acceleration. With this condition, the feasibility of the algorithm is not guaranteed as the region set for the admissible states is very restrictive in the last step.

Invariant conditions generate more possible solutions but by itself does not generate safe strategies. The "normal constraints" strategy refers to keeping a constant value for the constraints; it is the most undesirable strategy as it leaves a large number of unfeasible solutions. Adding the terminal P matrix for invariant conditions (S_2) and normal constraints (S_4) greatly improves the number of solutions but does not ensure a solution over the entire prediction horizon.

The strategy with the braking law (S_3) always finds a solution and consequently it is assumed that the cost function is a Lyapunov function ensuring closed loop stability. This is a consequence of a constant change in the restrictions directly to the control action, i.e., each sample is saturating the control.

In the simulations (Fig. 3) we can observe the specific force perceived in each scenario. Comparing this figure with Table III we can observe the trade off between performance and the number of feasible solutions.

Taking as an example only the second scenario, and evaluating the capabilities and limitations along x with respect to the working space, we can see in Fig. 4 that the braking law strategy does not completely maximize the workspace and also does not provide the most optimal response for the control and trajectory tracking. However, it is the only strategy that ensures closed-loop stability through feasibility. It should be noted that this depends greatly on the choice of the parameters in the control law, consequently the performance can be increased with the appropriate tuning.

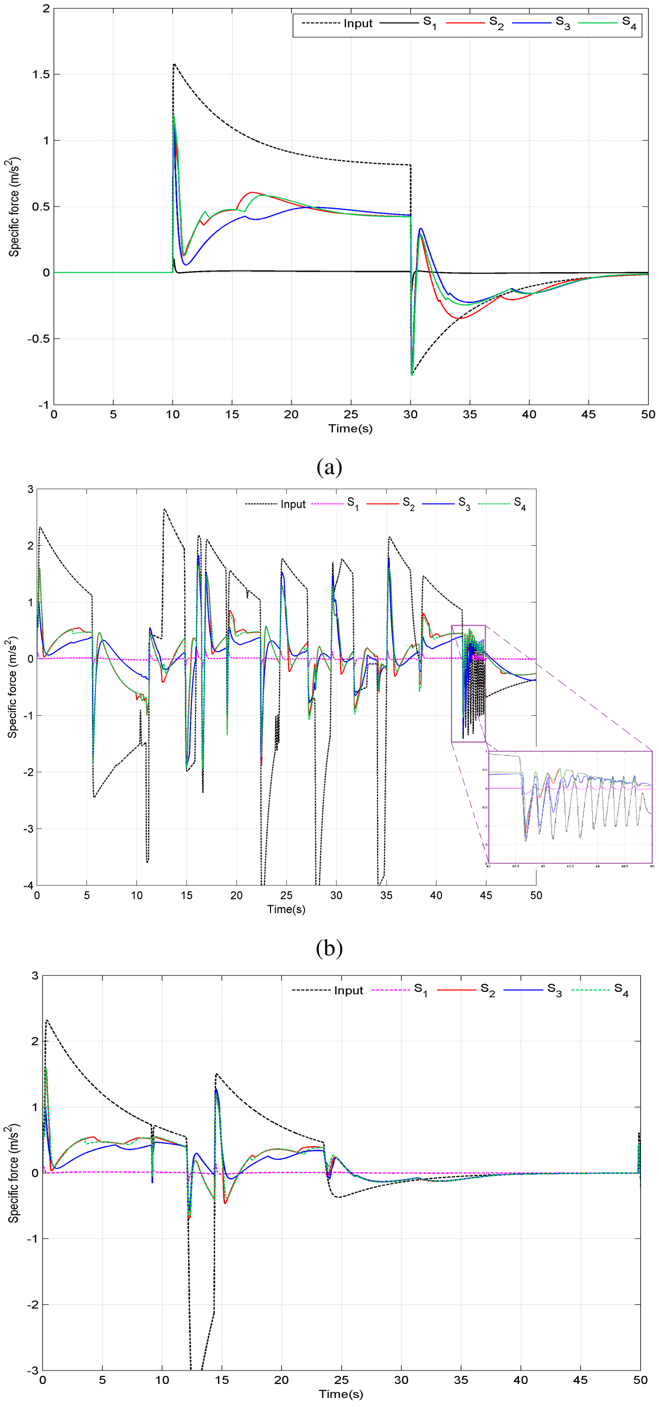

Fig. 3: Comparison of the simulator outputs for all different strategies in scenarios 1 (a), 2 (b) and 3 (c)

VI. CONCLUSION

In this paper different methodologies to handle feasibility issues and possible stability in MPC-based MCA schemes have been presented and discussed. A non-linear sampling frequency is implemented in the system to obtain a long enough prediction horizon and to achieve real-time execution.

Simulation results demonstrated that it is necessary to find a balance between maximizing the simulator's working space and the feasibility of the optimization problem. The different strategies compared in this paper reflect that it is recommended, if not necessary, to modify the way in which restrictions are posed at each step time in the optimization problem. If this is not possible, at least it is recommended to apply a strong weight at the end of the prediction horizon. Only an adequate constraint law applied directly in the control input provides conditions to guarantee feasibility and closed loop stability.

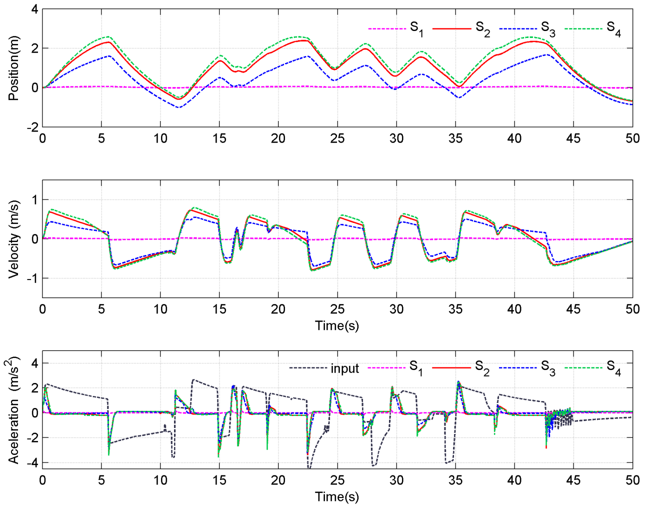

Fig. 4: Longitudinal motion comparison: position,velocity and acceleration in scenario 2

Future work will consist in performing experimental tests to validate the level of acceptability of the optimization method and the perception model used as the basis for prediction. Also we aim at modifying the braking law parameters in order to get optimal performance and ensure stability of the system even in the most critical driving situations.

APPENDIX

Consider the LTI system

$$\begin{aligned} x_m(k+1) &= A_m x_m(k) + B_m u(k) \\ y(k) &= C_m x_m(k) + D_m u(k) \end{aligned}$$

Then, the system is augmented to include an integral action in the plant model ensuring offset free tracking in the steady state [17]. Now, the model input Δu_k is the control increment instead of the control signal $u(k)$. In the next system, I_m represents the unit matrix.

$$\overbrace{\begin{bmatrix} x_m(k+1) \\ u(k) \end{bmatrix}}^{x_{k+1}} = \overbrace{\begin{bmatrix} A_m & B_m \\ 0_m & I_m \end{bmatrix}}^{A} \overbrace{\begin{bmatrix} x_m(k) \\ u(k-1) \end{bmatrix}}^{x_k} + \overbrace{\begin{bmatrix} B_m \\ I_m \end{bmatrix}}^{B} \Delta u_k$$

$$y_k = \overbrace{\begin{bmatrix} C_m & D_m \end{bmatrix}}^{C} \begin{bmatrix} x_m(k) \\ u(k-1) \end{bmatrix} + D\Delta u_k$$

REFERENCES

[1] W. Kwon and A. Pearson, “A modified quadratic cost problem and feedback stabilization of a linear system,” *IEEE Transactions on Automatic Control*, vol. 22, pp. 838–842, Oct. 1977.

[2] W. H. Kwon, A. M. Brucktein, and T. Kailath, “Stabilizing State-Feedback Desing Via The Moving Horizon Method,” in *21st IEEE Conference on Decision and Control*, pp. 234–239, 1982.

[3] M. Dagdelen, G. Reymond, A. Kemeny, B. M., and N. Mazi, “MPC Based motion cueing algorithm: developpement and application to the ULTIMATE driving simulator,” in *DSC 2004 Europe*, pp. 221–233, 2004.

[4] P.-O. Gutman and M. Cwikel, “An algorithm to find maximal state constraint sets for discrete-time linear dynamical systems with bounded controls and states,” *IEEE Transactions on Automatic Control*, vol. 32, pp. 251–254, Mar. 1987.

[5] Z. Fang and A. Kemeny, “Explicit MPC motion cueing algorithm for real-time driving simulator,” in *Power Electronics and Motion Control Conference (IPEMC), 2012 7th International*, vol. 2, pp. 874–878, IEEE, 2012.

[6] Z. Fang and A. Kemeny, “An efficient Model Predictive Control-based motion cueing algorithm for the driving simulator,” *SIMULATION*, vol. 92, pp. 1025–1033, Nov. 2016.

[7] F. Maran, M. Bruschetta, and A. Beghi, “Study of a real-time, MPC based motion cueing procedure with time-varying prediction for different classes of drivers,” in *American Control Conference (ACC)*, pp. 1711–1716, IEEE, July 2016.

[8] B. D. C. Augusto, “Motion cueing in the Chalmers driving simulator: An optimization-based control approach,” Master’s thesis, Technical University of Lisbon, 2009.

[9] A. Beghi, M. Bruschetta, and F. Maran, “A real time implementation of MPC based Motion Cueing strategy for driving simulators,” in *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*, pp. 6340–6345, IEEE, 2012.

[10] M. Baseggio, A. Beghi, M. Bruschetta, F. Maran, and D. Minen, “An MPC approach to the design of motion cueing algorithms for driving simulators,” in *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, pp. 692–697, IEEE, 2011.

[11] R. J. Telban and F. M. Cardullo, “Motion cueing algorithm development: Human-centered linear and nonlinear approaches,” Tech. Rep. CR-2005-213747, NASA, 2005.

[12] C. Rengifo, J.-R. Chardonnet, D. Paillot, H. Mohellebi, and A. Kemeny, “Solving the Constrained Problem in Model Predictive Control based Motion Cueing Algorithm with a Neural Network Approach,” in *17th Driving Simulation & Virtual Reality Conference & Exhibition*, pp. 63–69, Sept. 2018.

[13] C. Bordons and E. Camacho, *Model predictive control*. Springer Verlag London Limited, 2007.

[14] H. Ferreau, C. Kirches, A. Potschka, H. Bock, and M. Diehl, “qpOASES: A parametric active-set algorithm for quadratic programming,” *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, 2014.

[15] S.-H. Chen and L.-C. Fu, “An optimal washout filter design for a motion platform with senseless and angular scaling maneuvers,” in *American Control Conference (ACC), 2010*, pp. 4295–4300, IEEE, 2010.

[16] Z. Fang, M. Tsushima, E. Kitahara, N. Machida, D. Wautier, and A. Kemeny, “Motion cueing algorithm for high performance driving simulator using yaw table,” *IFAC-PapersOnLine*, vol. 50, pp. 15965–15970, July 2017.

[17] J. A. Rossiter, *Model-based predictive control: a practical approach*. Control series, Boca Raton: CRC Press, 2003.