Spatial Predictive Control for Agile Semi-Autonomous Ground Vehicles

Yiqi Gao*, Andrew Gray*, Janick V. Frasch†, Theresa Lin*, Eric Tseng‡, J. Karl Hedrick*, Francesco Borrelli*

***Department of Mechanical Engineering, University of California, Berkeley, USA**
† Interdisciplinary Center for Scientific Computing, University of Heidelberg. Heidelberg, Germany
‡ Ford Research Laboratories, Dearborn, MI, USA

Berkeley, CA 94720, USA
Phone: +1 510 6432044
E-mail: yiqigao@berkeley.edu

This paper presents a formulation to the obstacle avoidance problem for semi-autonomous ground vehicles. The planning and tracking problems have been divided into a two-level hierarchical controller. The high level solves a nonlinear model predictive control problem to generate a feasible and obstacle free path. It uses a nonlinear vehicle model and utilizes a coordinate transformation which uses vehicle position along a path as the independent variable. The low level uses a higher fidelity model and solves the MPC problem with a sequential quadratic programming approach to track the planned path. Simulations show the method's ability to safely avoid multiple obstacles while tracking the lane centerline. Experimental tests on a semi-autonomous passenger vehicle driving at high speed on ice show the effectiveness of the approach.

Topics / Vehicle Control, Autonomous Driving

1. INTRODUCTION

Real-time generation and tracking of feasible trajectories is a major barrier in autonomous guidance systems when the vehicle travels at the limits of its handling capability. Trajectories generated by using oversimplified models violate system constraints, while computing trajectories using high-fidelity vehicle models and nonlinear optimization is computationally demanding. Moreover, the presence of external disturbances and model uncertainties might still prevent the vehicle from following the desired path.

Because of its capability to systematically handle system nonlinearities and constraints, work in a wide operating region and close to the set boundary of admissible states and inputs, Model Predictive Control (MPC) is an attractive method to generate feasible trajectories and robustly track them [7]. Parallel advances in theory and computing systems have enlarged the range of applications where real-time MPC can be applied [3, 4, 14, 9, 16]. Yet, the computational burden of Nonlinear MPC (NMPC) is still a barrier for agile autonomous drive. Previous work by the authors [10] have accounted for pop-up obstacles by decomposing the problem into a two-level NMPC problem. The high-level path planner uses a simplified point-mass vehicle model to generate an obstacle avoiding trajectory by using a NMPC controller. The trajectory is fed to the low-level path follower designed by using a NMPC based on a higher fidelity vehicle model [7]. In [10] the proposed hierarchical framework has been implemented on an autonomous ground vehicle driving at high speeds on an icy road. Although this decomposition allows for real-time implementation, the trajectories generated by the point-mass path planner are not always feasible. The lower level tracking performance deteriorates and obstacle collisions can be observed in conditions where the obstacle could have been avoided. In order to overcome this issue and still maintain real-time feasibility, the work in [12] uses a motion primitive path planner at the high level. Although the high level trajectories from the motion primitive planner are feasible for the high-fidelity model, the optimal trajectory requires the online solution of a mixed-integer program or the offline computation of a large look-up table. This prevents the use of such approach on current electronic control units. In order to plan a feasible path in real-time this paper studies the use of *spatial predictive control at the high level*. We follow the approach presented in [15, 13] and transform time-dependent vehicle dynamics into spatial-dependent dynamics. By using this approach obstacle constraints are translated into spatial con-

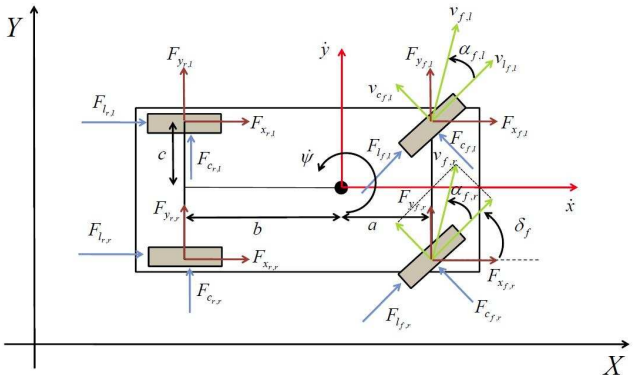


Fig. 1: Simplified Four Wheel Vehicle Dynamical Model.

straints on the state vector. Simulation and experimental results show the controller's ability to avoid multiple obstacles while tracking a reference in the center of the lane. Real-time tests have been conducted on a dSPACE platform in hardware-in-the-loop simulations. Furthermore, the hierarchical controller has been implemented on a semi-autonomous vehicle driving high-speed on ice.

The paper is structured as follows. Section 2 describes the vehicle models used in the controller, including a four wheel vehicle model and a spatial bicycle vehicle model. Section 3 outlines the hierarchical architecture of the controller and the MPC formulation. The simulation and experimental results are reported in section 4.

2. VEHICLE MODEL

This section introduces the vehicle models used for the control design. Section 2.1 describes the six state nonlinear vehicle model used in [7]. Section 2.2 explains the transformation from a time-dependent bicycle model to a spatial-dependent model.

2.1 Four Wheel Model

The vehicle dynamics described by the four wheel model [7] can be compactly written as

$$\dot{\xi}(t) = f^{4w}(\xi(t), u(t)), \tag{1}$$

where $\xi(t) \in \mathbb{R}^n$ is the state of the system and $u(t) \in \mathbb{R}^{m_r}$ is the input, $n = 6$ is the number of states and $m_r = 3$ is the number of inputs. The six states are lateral and longitudinal velocities in the body frame, the yaw angle, yaw rate, lateral and longitudinal vehicle coordinates in the inertial frame, $\xi = [\dot{y}, \dot{x}, \psi, \dot{\psi}, Y, X]^T$. The three inputs are $u = [\delta_f, F_{b_l}, F_{b_r}]^T$ where δ_f is the front steering angle and F_{b_l}, F_{b_r} are the left and right braking forces.

The dynamics in (1) can be derived by using the equations of motion about the vehicle's center of gravity (CoG) and coordinate transformations between the inertial frame and the vehicle body frame:

$$m\ddot{y} = -m\dot{x}\dot{\psi} + F_{y_{f,l}} + F_{y_{f,r}} + F_{y_{r,l}} + F_{y_{r,r}}, \tag{2a}$$

$$m\ddot{x} = m\dot{y}\dot{\psi} + F_{x_{f,l}} + F_{x_{f,r}} + F_{x_{r,l}} + F_{x_{r,r}}, \tag{2b}$$

$$I\ddot{\psi} = a(F_{y_{f,l}} + F_{y_{f,r}}) - b(F_{y_{r,l}} + F_{y_{r,r}}) + c(-F_{x_{f,l}} + F_{x_{f,r}} - F_{x_{r,l}} + F_{x_{r,r}}), \tag{2c}$$

$$\dot{Y} = \dot{x}\sin\psi + \dot{y}\cos\psi, \tag{2d}$$

$$\dot{X} = \dot{x}\cos\psi - \dot{y}\sin\psi, \tag{2e}$$

where the constant m is the vehicle's mass, I is the rotational inertia about the yaw axis, a and b are the distances from the CoG to the front and rear axles and c is the distance from the CoG to the left/right side at the wheels.

Figure 1 shows (1) the vehicle states, (2) the lateral (cornering) and longitudinal tire forces, $F_{c_{\star,\bullet}}$ and $F_{l_{\star,\bullet}}$, (3) the components of the tire forces along the lateral and longitudinal vehicle axes, $F_{y_{\star,\bullet}}$ and $F_{x_{\star,\bullet}}$ (4) the slip angle, $\alpha_{\star,\bullet}$ and (5) the steering angles, $\delta_\star$. The first subscript, $\star \in \{f, r\}$, denotes the front and rear, and the second subscript, $\bullet \in \{l, r\}$, denotes the left and right side of the vehicle.

The x and y components of lateral and longitudinal tire forces are

$$F_{y_{\star,\bullet}} = F_{l_{\star,\bullet}}\sin\delta_\star + F_{c_{\star,\bullet}}\cos\delta_\star, \tag{3a}$$

$$F_{x_{f,\bullet}} + F_{x_{r,\bullet}} = F_{b_\bullet}, \tag{3b}$$

where δ_f, F_{b_l} and F_{b_r} are the inputs to the system and δ_r assumed to be zero. The lateral and longitudinal tire forces, $F_{l_{\star,\bullet}}$ and $F_{c_{\star,\bullet}}$, are computed using a Pacejka tire model [2].

2.2 Spatial Bicycle Model

Figure 2 shows the curvilinear coordinate system used in the spatial model as well as the states of the model. The solid curve represents the *center line of the lane*. The states of the spatial vehicle model are defined as $\xi^s = [\dot{y},\ \dot{x},\ \dot{\psi},\ e_\psi,\ e_y]^T$. Where $\dot{y}$, $\dot{x}$ and $\dot{\psi}$ are body frame velocities, e_ψ and e_y are the error of heading angle and lateral position with respect to the center line of the lane.

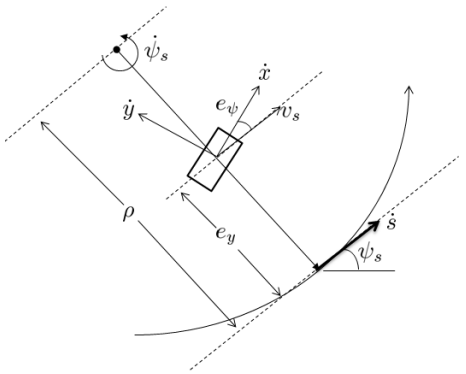


Fig. 2: The curvilinear coordinate system. The dynamics are derived about a curve defining the center-line of a track. The coordinate s defines the arc-length along the track. The relative spatial coordinates e_y and e_ψ are shown.

The following kinematic equations can be derived from Figure 2:

$$v_s = (\rho - e_y)\cdot\dot{\psi}_s; \quad v_s = \dot{x}\cdot cos(e_\psi) - \dot{y}\cdot sin(e_\psi), \tag{4}$$

where v_s is the projected vehicle speed along direction of the lane center line, ρ and ψ_s are the radius of curvature and the heading of the lane center line. $\dot{\psi}_s$ is the time derivative of ψ_s. The vehicle's velocity along the path $\dot{s} = \frac{ds}{dt}$ is then given by

$$\dot{s} = \rho \cdot \dot{\psi}_s \;=\; \frac{\rho}{\rho - e_y} \cdot (\dot{x} \cdot cos(e_\psi) - \dot{y} \cdot sin(e_\psi)). \quad (5)$$

where s is the projected vehicle position along the lane center line. Using simple relationships in the new curvilinear coordinate system and the fact that $\frac{d\xi^s}{ds} = \frac{d\xi^s}{dt} \cdot \frac{dt}{ds}$ we can calculate the derivative of ξ^s with respect to s as follows ($(\cdot)'$ represents the derivative with respect to s):

$$\dot{y}' = \ddot{y}/\dot{s}; \qquad \dot{x}' = \ddot{x}/\dot{s}; \qquad \dot{\psi}' = \ddot{\psi}/\dot{s}; \quad (6a)$$

$$e'_\psi = (\psi - \psi_s)' = \dot{\psi}/\dot{s} - \psi'_s; \quad (6b)$$

$$e'_y = \dot{e}_y/\dot{s} = (\dot{x} \cdot sin(e_\psi) + \dot{y} \cdot cos(e_\psi))/\dot{s} \quad (6c)$$

where $\ddot{y}, \ddot{x}$ and $\ddot{\psi}$ are computed from a bicycle model [5], a simplified version of the four wheel model (2). The road information ψ'_s is assumed to be known. The spatial vehicle dynamics then can be formulated as:

$$\xi^{s'}(s) = f^s(\xi^s(s), u^s(s)). \quad (7)$$

where the inputs are the front steering angle δ_f and the braking or throttle effort $\beta_r \in [-1\ 1]$ with -1 corresponding to maximum braking and 1 corresponding to maximum throttle ($u^s = [\delta_f, \beta_r]$). Note the time as function of s, $t(s)$, can be retrieved by integrating $t' = 1/\dot{s}$.

3. MPC CONTROL ARCHITECTURE WITH OBSTACLE AVOIDANCE

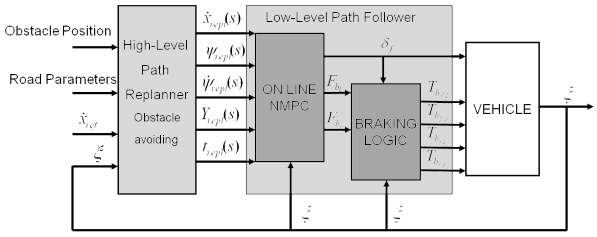


Fig. 3: Architecture of the two-level MPC. A spatial vehicle model is used for high-level path planning. A four-wheel vehicle model is used for low-level path tracking.

A two level hierarchical control scheme [10] is used in this paper. As illustrated in Fig 3, the obstacle avoiding control problem is decomposed into two parts: a high level path planner and a low level path follower. The road information is fed into the high level. The high level plans an obstacle-free path using the nonlinear spatial vehicle model in section 2.2. The planned path (in spatial coordinates) is then fed to the low level path follower, which uses the four-wheel vehicle model (section 2.1) to compute the optimal control inputs in order to track the planned path. Each level solves a nonlinear MPC problem to plan or follow the path.

3.1 MPC Formulation

The MPC controller is formulated as a general optimization problem

$$\min_{U_t} \quad J_N(\tilde{\xi}_t, U_t, \Delta U_t) \quad (8a)$$

$$\text{subj. to} \quad \xi_{k+1,t} = f(\xi_{k,t}, u_{k,t}) \quad k = t, ..., t + H_p - 1 \quad (8b)$$

$$\Delta u_{k+1,t} = u_{k+1,t} - u_{k,t} \quad k = t, ..., t + H_p - 1 \quad (8c)$$

$$\xi_{k,t} \in \mathcal{X} \quad k = t + 1, ..., t + H_p - 1 \quad (8d)$$

$$u_{k,t} \in \mathcal{U} \quad k = t, ..., t + H_p - 1 \quad (8e)$$

$$\Delta u_{k,t} \in \Delta\mathcal{U} \quad k = t + 1, ..., t + H_p - 1 \quad (8f)$$

$$\xi_{t,t} = \xi(t) \quad (8g)$$

where $\tilde{\xi}_t = [\xi_{t,t}\ ,\ \xi_{t+1,t}\ , ...,\ \xi_{t+H_p-1,t}]$ is the sequence of states $\tilde{\xi}_t \in \mathbb{R}^{nH_p}$ over the prediction horizon H_p predicted at time t. Elements in $\tilde{\xi}_t$ are updated according to the discretized dynamics of the vehicle model (8b). $u_{k,t}$ and $\Delta u_{k,t} \in \mathbb{R}^{m_r}$ is the k^{th} vector of the input sequence $U_t \in \mathbb{R}^{m_r H_p}$ and $\Delta U_t \in \mathbb{R}^{m_r(H_p-1)}$ respectively,

$$U_t = [u^T_{t,t}\ ,\ u^T_{t+1,t}\ , ...,\ u^T_{t+H_u-1,t}\ ,\ u^T_{t+H_u,t}\ , ...,\ u^T_{t+H_p-1,t}]^T \quad (9a)$$

$$\Delta U_t = [\Delta u^T_{t+1,t}, ..., \Delta u^T_{t+H_u-1,t}, \Delta u^T_{t+H_u,t}, ..., \Delta u^T_{t+H_p-1,t}]^T \quad (9b)$$

Note the time "t" in this section denotes the *generalized time*, the independent variable with respect to which the system states are differentiated. Specifically, in the high level it is the spatial coordinate "s" while in the low level it is the time t.

The computational complexity of the MPC problem is reduced by two means. One is to hold the last $H_p - H_u$ input vectors in U_t constant and equal to the vector $u_{t+H_u-1,t}$. With this assumption, only the first H_u input vectors constitutes the optimization variables. The other one is to keep the input vectors constant for every iH_u steps in the first H_u input vectors (i.e., set $u_{t+n\cdot iH_u+k,t} = u_{t+n\cdot iH_u,t}, \forall k = 1, 2...iH_u$). With this assumption, the number of optimization variables is further reduced. We refer to H_p as the *prediction horizon*, H_u as the *control horizon* and iH_u as the *input blocking factor*.

At each time step t, the performance index $J_N(\tilde{\xi}_t, U_t, \Delta U_t)$ is optimized under the constraints (8c)-(8f) starting from the state $\xi_{t,t} = \xi(t)$ to obtain an optimal control sequence, $U^*_t = [{u^*_{t,t}}^T, ..., {u^*_{t+H_p-1,t}}^T]^T$. The first of such optimal moves $u^*_{t,t}$ is the control action applied to the vehicle at time t. At time $t+1$, a new optimization is solved over a shifted prediction horizon starting from the new measured state $\xi_{t+1,t+1} = \xi(t+1)$. The time interval between time step $t+1$ and time step t is the sampling time T_s.

3.2 High Level Path Planner

The High level MPC uses the spatial bicycle vehicle model. Specifically, in equation (8b) we use a discretized version of the equation 7.

The cost function considers the deviation of the tracking states $\eta^{hl}_{k,s} = [\dot{x}_{k,s}\ , \dot{\psi}_{k,s},\ e_{\psi_{k,s}}, e_{y_{k,s}}]^T$ with respect to the reference $\eta^{hl}_{ref_{k,s}} = [\dot{x}_{k,s}\ , \dot{\psi}_{k,s},\ e_{\psi_{k,s}},\ e_{y_{k,s}}]^T$, as well as the input and input rate.

$$J_N(\tilde{\xi}_s, U_s, \Delta U_s) = \sum_{k=s}^{s+H_{p,hl}-1} \|\eta^{hl}_{k,s} - \eta^{hl}_{ref_{k,s}}\|^2_{Q_{hl}} + \\ + \|u_{k,s}\|^2_{R_{hl}} + \|\Delta u_{k,s}\|^2_{S_{hl}} \tag{10}$$

Where the reference $\dot{x}_{ref_{k,s}}$ is a given constant, $e_{\psi_{ref_{k,s}}}$ and $e_{y_{ref_{k,s}}}$ are zero, $\dot{\psi}_{ref_{k,s}}$ is defined as $\psi'_s \cdot \dot{x}_{ref_{k,s}}$. Q_{hl}, R_{hl} and S_{hl} are weighting matrices with proper dimensions. The index for *general time* "t" is replaced by "s" here. The distance interval between step $s+1$ and s is the sampling interval ds.

A spatial horizon allows one to formulate obstacle constraints as simple bounds on e_y and include them in the state constraints (8d). At each prediction step, the vehicle position along the lane center is known to be $(s+k\cdot ds)$. According to the position and width of the obstacle, one can determine the bounds on e_y. With one obstacle, there are two disconnected regions of feasible e_y, respectively corresponding to passing the obstacle from left and right. In this paper, a simple heuristic based on the vehicle position and the size of each feasible region is used to determine which side should the vehicle pass through. In the case of multiple obstacles at the same coordinate "s", a similar approach can still be used.

The optimal trajectory $[\dot{x}^*(s), \dot{\psi}^*(s), e^*_{\psi}(s), e^*_y(s), t^*(s)]$ is computed by simulating the vehicle model with the optimal inputs from the MPC problem and then passed to the low level path follower. Note $t^*(s)$ can be easily retrieved as described in section 2.2. In the low level, this spatial trajectory is transformed back to a time-dependent trajectory $[\dot{x}^*(t), \dot{\psi}^*(t),\ \psi^*(t),\ Y^*(t),\ X^*(t)]$ by coordinate transformation and interpolation.

3.3 Low Level Path Follower

The low level MPC uses the four wheel vehicle model from section . At each time step t, the system dynamics are linearized around the equilibrium trajectory $[\xi_{k,t}, u_{k,t}]$, with $u_{k,t} = u_{t,t}\ \forall k = t, ..., t + H_{p,ll}$ and $\xi_{k+1,t} = f^{4w}_d(\xi_{k,t}, u_{k,t})$, where f^{4w}_d is the discrete version of the equation $\dot{\xi}(t) = f^{4w}(\xi(t), u(t))$. The details of the linearizing process can be found in [6].

The cost function again consists of the deviation of the tracking states $\eta^{ll}_{k,t} = [\dot{x}_{k,t}, \psi_{k,t},\ \dot{\psi}_{k,t}, Y_{k,t}]^T$ from the reference $\eta^{ll}_{ref_{k,t}} = [\dot{x}_{ref_{k,t}},\ \psi_{ref_{k,t}},\ \dot{\psi}_{ref_{k,t}},\ Y_{ref_{k,t}}]^T$ as well as the input and input rate penalty.

$$J_N(\tilde{\xi}_t, U_t, \Delta U_t) = \sum_{k=t}^{s+H_{p,ll}-1} \|\eta^{ll}_{k,t} - \eta^{ll}_{ref_{k,t}}\|^2_{Q_{ll}} + \\ \|u_{k,t}\|^2_{R_{ll}} + \|\Delta u_{k,t}\|^2_{S_{ll}} \tag{11}$$

The inputs vector $u_{k,t} = [\delta_f, F_{b_l}, F_{b_r}]^T$ consists of the steering angle δ_f, left braking force F_{r_l} and right braking force F_{b_r}. The braking logic in [7] is used to distribute the corresponding torques at the four wheels.

4. SIMULATION AND EXPERIMENTAL RESULTS

Simulation and experimental tests are conducted to evaluate the proposed controller. The MPC problem has been implemented as a C-coded s-function where NPSOL [11] is used to solve the high level optimization. At the low-level, the nonlinear program is solved by using a sequential quadratic programming approach [8] and the quadratic program is solved using the QP solver routine available in [1], which implements the Dantzig-Wolfe's algorithm.

4.1 Simulation Setup Description

Real-time simulation of the controller is tested on a dSPACE rapid prototyping system consisting of a MicroAutoBox and a DS1006 processor board with a DS2211 I/O board. The controller runs on the MicroAutoBox. The first element of the optimal control sequence is passed to DS1006 board, which simulates the vehicle dynamics using a four wheel vehicle model and Pacejka tire model [2], and then feeds the current vehicle state back to the controller. The two components communicate through a CAN bus.

4.2 Experimental Setup Description

The experiments have been performed at a test center equipped with icy and snowy handling tracks. The MPC controller has been tested on a passenger car, with a mass of 2050 Kg and a yaw inertia of 3344 Kg/m^2. The controllers were run in a dSPACE Autobox system, equipped with a DS1005 processor board and a DS2210 I/O board.

We used an Oxford Technical Solution (OTS) RT3002 sensing system to measure the vehicle position and orientation in the inertial frame and the vehicle velocities in the vehicle body frame. The OTS RT3002 is housed in a small package that contains a differential GPS receiver, Inertial Measurement Unit (IMU) and a DSP. The IMU includes three accelerometers and three angular rate sensors. The DSP receives both the measurements from the IMU and the GPS, utilizes a Kalman filter for sensor fusion, and calculates the position, orientation and other states of the vehicle.

The car was equipped with an Active Front Steering (AFS) and Differential Braking system which utilizes an electric drive motor to change the relation between the hand steering wheel and road wheel angles. This is done independently from the hand wheel position, thus the front road wheel angle is obtained by summing the driver hand wheel position and the actuator angular movement.

The sensor, the dSPACE Autobox and the actuators communicate through a CAN bus.

4.3 Results and Discussions

The controller is tested with the following tuning: **Tuning HL** *Hierarchical MPC High-level:* Nonlinear

MPC (8) with model (7) and cost (10) and with the following parameters.

- $ds = 1.5\text{m}$, $H_{p,hl} = 15$, $H_{u,hl} = 12$, $iH_{u,hl} = 3$
- $\delta_f \in [-10\,^\circ, 10\,^\circ]$, $\Delta\delta_f \in [-17\,^\circ, 17\,^\circ] \times d_s$
- $\beta_r \in [-1, 1]$, $\Delta\beta_r \in [-10, 10] \times d_s$
- $Q_{hl} = \text{diag}(1, 1, 20, 1)$, $\quad R_{hl} = \text{diag}(50, 50)$
  $S_{hl} = \text{diag}(0.1, 0.1)$

Tuning LL *Hierarchical MPC Low-level:* Nonlinear MPC (8) with model (1) and cost (11) and with the following parameters

- $T_{s,ll} = 0.05\text{s}$, $H_{p,ll} = 5$, $H_{u,ll} = 3$, $iH_{u,ll} = 1$
- $\delta_f \in [-10\,^\circ, 10\,^\circ]$, $\Delta\delta_f \in [-17\,^\circ, 17\,^\circ] \times T_{s,ll}$
- $F_{b_\bullet} \in [-1500, 0]$, $\Delta F_{b_\bullet} \in [-1000, 1000] \times T_{s,ll}$
- $Q = \text{diag}(10, 20, 10, 50)$, $\quad R = \text{diag}(1, 0.5, 0.5)$
  $S = \text{diag}(1, 0.1, 0.1)$

The simulation and experimental results are summarized in Figures 4 to 7. In all tests, the road friction coefficient is approximately 0.3. The high level path planner is invoked every 200ms, and the low level every 50ms.

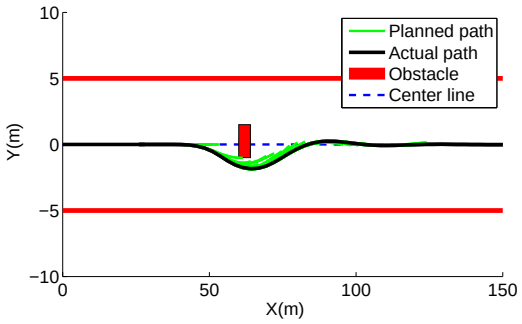


Fig. 4: **Simulated result**. The vehicle entered the maneuver at 50 kph. The green lines are planned paths from the high-level which are updated every 200 ms. The black line is the actual trajectory the vehicle traveled.

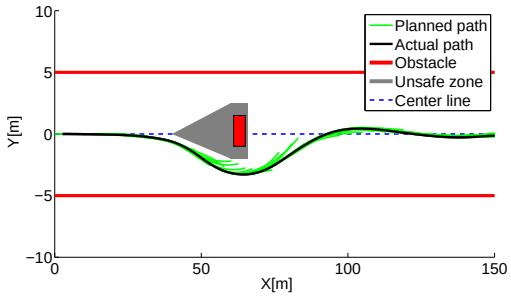


Fig. 5: **Experimental result**. The vehicle entered the maneuver at 50 kph. Friction coefficient of the ground was approximately 0.3. The vehicle avoided the obstacle and continued to track the lane center. The green lines are planned paths from the high-level which were updated every 200 ms. The black line is the actual trajectory the vehicle traveled.

Since the obstacle constraint is formulated as hard constraints on the states, the high level path planner will always plan a tight path passing the obstacle due to the tracking error and input penalties, see the green line right below the obstacle in figure 4 as an example. To avoid this problem, an unsafe zone around the obstacle is added in the tests. The unsafe zones are shown in gray in figure 5 to 7.

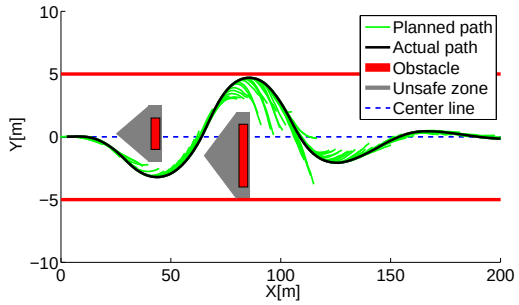


Fig. 6: **Experimental result**. The vehicle entered the maneuver at 50 kph. Friction coefficient of the ground was approximately 0.3. The vehicle avoided the two obstacles and continued to track the lane center.

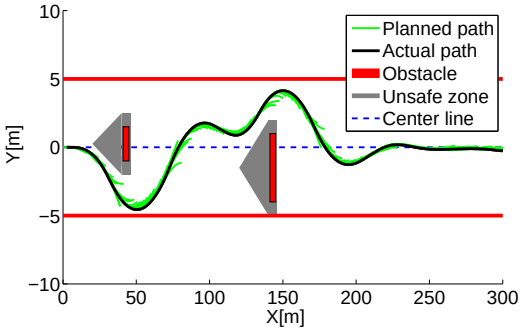
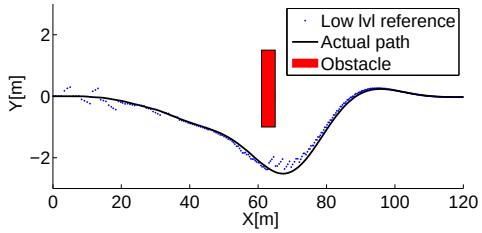


Fig. 7: **Experimental result**. The vehicle entered the maneuver at 50 kph. Friction coefficient of the ground was approximately 0.3. The vehicle avoided the first obstacle and continued to track the lane center until the second obstacle came into sight. It then turned left to avoid the second obstacle.

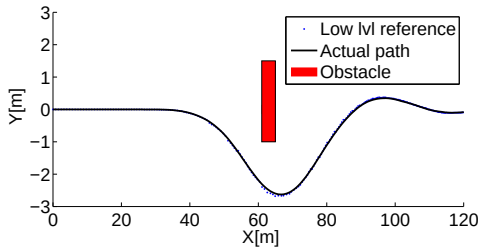
Figure 4 and 5 are the simulation and experimental results of the vehicle avoiding one obstacle on a slippery road. The two tests show consistent performance. Figure 6 and 7 show the experimental results for avoiding two obstacles with different distances between them. In both cases the vehicle was able to avoid both obstacles and get back to the lane center afterwards. In figure 7 when the distance between the two obstacles was large, the vehicle was already trying to get back to the lane center before the second obstacle came into the planning horizon.

Figure 8 compares the controller's performance with a previously proposed controller in [10], which uses a time-dependent point mass model at the high level. The blue dots in the figures are the reference for the low level at each sampling time. We observe that the use of a nonlinear bicycle model in the high level greatly improves the tracking of low level to the high level.

The use of a spatial model in the high level makes the construction of obstacle constraints straight forward as described in section 3.1 . On the other hand, the hard constraint formulation for obstacle is not trivial for general shaped obstacles in MPC formulation with time-dependent models. Potential field ap-



(a) Simulation result of the controller in [10]. A time-dependent point mass model is used for the high level. The vehicle turns early because the high level uses a potential field approach for obstacle avoidance.



(b) Simulation result of the new controller. A spatial-dependent bicycle model is used for the high level.

Fig. 8: Simulation result comparison with the controller proposed in [10]. In both cases the high levels replan every 200ms. The same low level path follower is used, which uses a nonlinear four wheel model and runs every 50ms.

proaches are common in those situations. Of course, it is possible to find indicating functions whose level sets can be used to form the obstacle constraints. This is usually not straight forward, especially if one wants the indicating function to have nice properties, such as differentiability, in order to speed up the optimization problem.

5. CONCLUSION

In this paper, a hierarchical design of a MPC controller is presented for the obstacle avoiding and lane keeping problem of a ground vehicle on a slippery road. The control problem is decomposed into a high level path planner and a low level path follower. The high level uses a simplified spatial model while the low level uses a higher fidelity model. Experimental results show the controller is real-time implementable and able to avoid multiple obstacles at high speeds on slippery road.

REFERENCES

[1] *Model predictive control toolbox*. The MathWorks, Inc., Natick, MA, 2005.

[2] E. Bakker, L. Nyborg, and H. B. Pacejka. Tyre modeling for use in vehicle dynamics studies. *SAE paper # 870421*, 1987.

[3] F. Borrelli, A. Bemporad, M. Fodor, and D. Hrovat. An MPC/hybrid system approach to traction control. *IEEE Trans. Control Systems Technology*, 14(3):541–552, May 2006.

[4] F. Borrelli, T. Keviczky, G. J. Balas, G. Stewart, K. Fregene, and D. Godbole. Hybrid decentralized control of large scale systems. In *Hybrid Systems: Computation and Control*, Lecture Notes in Computer Science. Springer Verlag, March 2005.

[5] P. Falcone. *Nonlinear Model Predictive Control for Autonomous Vehicles*. PhD thesis, Universitá del Sannio, Dipartimento di Ingegneria, Benevento, Italy, June 2007.

[6] P. Falcone, F. Borrelli, J. Asgari, and E. Tseng. Linear time-varying model predictive control and its application to active steering systems: Stability analysis and experimental validation. *International Journal of Robust and Nonlinear Control*, 18:862–875, 2007.

[7] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat. Low complexity mpc schemes for integrated vehicle dynamics control problems. *9th International Symposium on Advanced Vehicle Control*, 2008.

[8] P. Falcone, E. Tufo, F. Borrelli, J. Asgari, and E. Tseng. Linear time varying model predictive control approach to the integrated vehicle dynamics control problem in autonomous systems. *IEEE Conference on Decision and Control*, 46, 2007.

[9] H. J. Ferrau, H. G. Bock, and Moritz Diehl. An online active set strategy for fast parametric quadtratic programming in mpc applications. *IFAC Workshop on Nonlinear Model Predictive Control for Fast Systems, plenary talk*, 2006.

[10] Y. Gao, T. Lin, F. Borrelli, E. Tseng, and D. Hrovat. Predictive control of autonomous ground vehicles with obstacle avoidance on slippery roads. *Dynamic Systems and Control Conference, 2010*, 2010.

[11] P. Gill, W. Murray, M. Saunders, and M. Wright. NPSOL – Nonlinear Programming Software. Stanford Business Software, Inc., Mountain View, CA, 1998.

[12] A. Gray, Y. Gao, T. Lin, J.K. Hedrick, E. Tseng, and F. Borrelli. Predictive control for agile semi-autonomous ground vehicles using motion primitves. *American Contr. Conf.*, 2012.

[13] F. Kehrle, J.V. Frasch, C. Kirches, and S. Sager. Optimal control of formula 1 race cars in a vdrift based virtual environment. In *IFAC World Congress Milan*, 2011.

[14] G. J. Balas T. Keviczky. Flight test of a receding horizon controller for autonomous uav guidance. In *Proc. American Contr. Conf.*, 2005.

[15] D. Verscheure, B. Demeulenaere, J. Swevers, J. De Schutter, and M. Diehl. Time-optimal path tracking for robots: A convex optimization approach. *Automatic Control, IEEE Transactions on*, 54(10):2318 –2327, oct. 2009.

[16] V. M. Zavala, C. D. Laird, and L. T. Biegler. Fast solvers and rigorous models: Can both be accomodated in nmpc. *IFAC Workshop on Nonlinear Model Predictive Control for Fast Systems, plenary talk*, 2006.