Coordinating Tire Forces to Avoid Obstacles Using Nonlinear Model Predictive Control

Matthew Brown and J. Christian Gerdes

***Abstract*—In order to safely navigate highly dynamic scenarios, automated vehicles must be able to react quickly to changes in the environment and be able to understand trade-offs between lateral and longitudinal forces when limited by tire-road friction. We present a design and experimental validation of a nonlinear model predictive controller that is capable of handling these complex situations. By carefully selecting the vehicle model and mathematical encodings of the vehicle and obstacles, we enable the controller to quickly compute inputs while maintaining an accurate model of the vehicle's motion and its proximity to obstacles. Experimental results of a test vehicle performing an emergency double lane change to avoid two "pop-up" obstacles demonstrate the ability of the controller to coordinate lateral and longitudinal tire forces even in emergency situations when the tires are at their friction limits.**

***Index Terms*—Automated vehicles, vehicle dynamics, obstacle avoidance, nonlinear model predictive control.**

I. Introduction

As AUTOMATED vehicles become more developed, they will need to handle a wide range of real-world situations. There are many challenging problems to overcome in order to keep the vehicle and other nearby stakeholders safe in complex urban environments, even when assuming low speeds and accelerations. Beyond these, automated vehicles must also handle dynamic situations at higher speeds and accelerations. In particular, it may be necessary to take sudden and extreme actions to avoid collision with another vehicle or object. In emergency situations, we want to fully utilize the vehicle's capability in order to avoid collisions.

Model predictive control (MPC) has become a popular technique that combines the predictive power of motion planning with the speed and robustness of real-time control. It enables the vehicle to act quickly to changes in the environment while ensuring that actions taken now will not put the vehicle in a dangerous state in the future. At each time step, a model predictive controller solves an optimization problem to compute a trajectory of states and inputs. It applies the first input (or some initial input sequence), and repeats at the next time step. Often MPC controllers are formulated as tracking controllers, where the objective is to follow a desired state trajectory (computed by a higher level motion planner or decision maker) while respecting constraints like input saturation and collision avoidance. Each of these controllers represents a collection of design decisions about the dynamics model, obstacle representation, and numerical approximations that make appropriate tradeoffs between model fidelity, optimality, and computation time.

Linear, or linear time varying, MPC has proven to be very effective at controlling a vehicle in situations where fast solve times are especially important. By solving a convex optimization problem with a linear dynamics model, a controller can very quickly find the global optimal of an approximate problem. Falcone *et al.* [1] demonstrated the effectiveness of linear MPC as a path tracker on icy roads at high speeds and ensured yaw stability of the vehicle with an additional constraint [2]. Similarly, Brown *et al.* [3] and Funke *et al.* [4] enforced yaw stability of the vehicle on a dry asphalt road by constraining velocity states to remain within a known safe region of the state space, defined by Beal and Gerdes [5]. The predictive power of the model is crucial for these tasks, ensuring that inputs applied now will stabilize the vehicle without compromising future performance. Using linear models enables the fast performance needed for stabilization, but also limits the controller's capabilities. In particular, lateral and longitudinal dynamics are coupled in a nonlinear way, making it challenging for a linear MPC controller to plan both lateral and longitudinal inputs. This is especially true when the tires are pushed to their friction limits and the system becomes more nonlinear and difficult to approximate with a linear model.

An alternative approach is to use a nonlinear vehicle model and solve a nonlinear optimization problem. This trades the global optimality and convergence guarantees of convex optimization for additional modeling power. Falcone *et al.* [1] used nonlinear MPC (NMPC) to compute steer angles to track a path during a double lane change and analyzed how incorporating braking forces affected the problem complexity [6]. They presented a controller that modeled the total forces on each tire but noted that the model complexity limited real-time implementations. Liu *et al.* [7] and Febbo *et al.* [8] investigated obstacle avoidance tasks, planning steering angle and a reference speed profile every 0.5 s. While these algorithms were able to maneuver through complex environments using nonlinear models, the long solve times limit fast reactions to changes in

Manuscript received August 8, 2018; revised April 4, 2019; accepted July 5, 2019. Date of publication November 22, 2019; date of current version February 25, 2020. This work is supported in part by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-1147470 and by Renault S.A. *(Corresponding author: Matthew Brown.)*

The authors are with the Department of Mechanical Engineering, Stanford University, Stanford, CA 94305 USA (e-mail: mjbrown@stanford.edu; gerdes@stanford.edu).

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIV.2019.2955362

the environment. Despite these challenges, nonlinear models are attractive because of their ability to capture the coupling between lateral and longitudinal forces on a tire.

Others have investigated methods for mitigating the long solve times of NMPC; Gao *et al.* [9] compared directly solving for steering and braking commands to a hierarchical approach that first found an obstacle-free path using a lower fidelity model before solving for steering and brake commands. In particular, they noted the challenges of combining obstacle avoidance and vehicle control in a single planner/controller that can run in real time. Frasch *et al.* approached the computation time problem of NMPC with real-time iteration and presented simulation results of a vehicle avoiding obstacles on a low friction surface in a structured environment [10]. We chose to focus on a single-level controller that fully converges before inputs are used, and to mitigate the long solve times of NMPC with our choice of integrator, variables, and replanning strategy.

A key design decision for an obstacle avoidance task is the representation of the vehicle and obstacles. Simple bounds on lateral position are amenable to fast solve times but are limited in modeling vehicle rotation and are often coupled with longitudinal position [3], [10]. More complex representations that explicitly consider distance to obstacles do not suffer these drawbacks, but take longer to solve. A common approach is to represent both the vehicle (or robot) and obstacles with sets of polytopes and to incorporate distance between these sets into the objective or constraints [9], [11], [12]. While the distance function is typically not differentiable, there are reasonable heuristics to address this. Gerdts *et al.* [13] took a different approach by modeling the robot and obstacles as unions of convex polyhedra and invoking Farkas' lemma to transform the constraint that the robot and obstacle do not overlap into something tractable for numerical optimization. Zhang *et al.* [14] extended this idea to include the notion of signed distance, to enable computation of minimum-penetration trajectories. This method allows robots and obstacles to be modeled as polytopes and avoids an often non-differentiable signed distance function, but it does introduce additional variables and constraints to the problem. Because these additions must be added to each stage, they can cause a meaningful increase in solve times. We represent the vehicle with a set of circles, similar to Zieglar *et al.* [12], and extend this idea by representing obstacles with another set of circles. This representation enables fast computation of the value and gradient of the signed distance function between the vehicle and an obstacle, and by using multiple circles can still capture the important effect of vehicle rotation.

In emergency situations, a controller needs to act quickly, have a sufficient understanding of the trade-offs necessary between lateral and longitudinal forces at the tires, and use a vehicle and obstacle representation that is valid for large vehicle rotations. The design decisions necessary to achieve this, especially on an experimental vehicle, represent a contribution to the literature. We propose a NMPC controller to compute the inputs of steer angle and front and rear longitudinal forces. The vehicle is modeled by a single track dynamics model with a brush tire model, accounting for steady state longitudinal weight transfer and derating lateral force due to longitudinal force. Obstacle avoidance is handled by a novel representation of the vehicle and any obstacles using sets of circles and penalizing the signed distance between all vehicle and obstacle circle pairs. The choice of total longitudinal force and brake bias as optimization variables reduces the problem size while still enabling the controller to arbitrarily direct braking force between the front and rear axles. This careful problem formulation enables a controller that is fast enough to run in real time, replanning every 50 ms, and sophisticated enough to operate at the vehicle's limits, coordinating lateral and longitudinal forces with one or more tires fully saturated. Section II describes the chosen vehicle model, Section III explains the representation of the vehicle and obstacles, and Section IV describes the formulation of the optimization problem. Section V compares solve times for controllers using different vehicle and obstacle representations and demonstrates the efficacy of the controller on a full-size vehicle, performing a sufficiently extreme maneuver to justify the chosen model complexity. In the presented experimental results, two obstacles "pop up" in front of the vehicle, forcing it to perform an emergency double lane change. The controller coordinates braking and turning forces of each axle individually while successfully avoiding the obstacles. Video of this experiment is available at *https://purl.stanford.edu/kw432sz0082*.

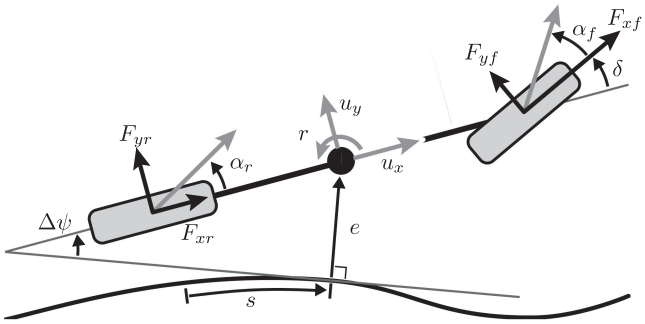


Fig. 1. The single track model positioned within a curvilinear coordinate system defined by a reference line.

II. Vehicle Model

The vehicle motion can be predicted with a planar single-track model with tire forces coming from a single-friction-coefficient brush tire model.

A. Curvilinear Coordinate System

The position of the vehicle is measured relative to a reference line. This line is assumed to have continuous curvature κ, which can be integrated to compute heading and position of the reference line. While this line can define an arbitrary coordinate system, in this paper we assume that the line corresponds to the center of the road. The vehicle's position is parameterized by relative heading $\Delta\psi$, longitudinal position s, and lateral position e.

B. Equations of Motion

A schematic of the single track model, or bicycle model, is shown in Fig. 1. The velocity variables of the vehicle are the yaw rate r and the lateral and longitudinal velocity at the center

of mass (CoM) u_y and u_x, respectively. We are interested in explicitly modeling the lateral and longitudinal forces on each axle; the equations of motion in terms of these front and rear lateral forces F_{yf}, F_{yr}, front and rear longitudinal forces F_{xf}, F_{xr}, and steering angle δ are:

$$\begin{aligned}
\dot{r} &= \frac{1}{I_{zz}}\left(aF_{yf}\cos\delta + aF_{xf}\sin\delta - bF_{yr}\right)\\
\dot{u}_y &= \frac{1}{m}\left(F_{yf}\cos\delta + F_{xf}\sin\delta + F_{yr}\right) - ru_x\\
\dot{u}_x &= \frac{1}{m}\left(-F_{yf}\sin\delta + F_{xf}\cos\delta + F_{xr} - F_{x\text{drag}}\right) + ru_y\\
\Delta\dot{\psi} &= r - \kappa\dot{s}\\
\dot{s} &= \frac{u_x\cos\Delta\psi - u_y\sin\Delta\psi}{1-\kappa e}\\
\dot{e} &= u_x\sin\Delta\psi + u_y\cos\Delta\psi
\end{aligned} \tag{1}$$

A simple drag model appropriate for operation below highway speeds is used: $F_{x\text{drag}} = C_{d0} + C_{d1}u_x$, but a quadratic term for aerodynamic effects at higher speeds could be included without changing the structure of the problem.

C. Tire Model

The lateral forces on the front and rear tires are approximated with a single-friction brush tire model. This model predicts lateral force as a function of the slip angle α, the angle between the tire's orientation and its velocity vector. This can be computed as:

$$\alpha_f = \tan^{-1}\left(\frac{u_y + ar}{u_x}\right) - \delta \tag{2}$$

$$\alpha_r = \tan^{-1}\left(\frac{u_y - br}{u_x}\right) \tag{3}$$

While there is a pleasant symmetry in using a combined slip model to capture the interaction between longitudinal and lateral forces, the fast wheel speed dynamics can make working with longitudinal slip difficult. For this reason, we approximate the combined slip brush tire model by derating the maximum lateral force $F_{y,\text{MAX}}$ with the given longitudinal force according to the friction circle, as presented by Hindiyeh [15], derived from Pacejka [16]. Lateral force is therefore a function of the lateral slip angle α, the longitudinal force F_x (assumed as an input), the normal load F_z, and the parameters cornering stiffness C_α and friction μ.

$$F_{y,\max} = \sqrt{(\mu F_z)^2 - {F_x}^2} \tag{4}$$

$$F_y = \begin{cases} -C_\alpha\tan\alpha + \frac{C_\alpha^2}{3F_{y,\max}}|\tan\alpha|\tan\alpha\ldots \\ -\frac{C_\alpha^3}{27(F_{y,\max})^2}\tan^3\alpha, & |\alpha| < \tan^{-1}\left(\frac{3F_{y,\max}}{C_\alpha}\right) \\ -F_{y,\max}\ \text{sgn}\,\alpha, & \text{otherwise} \end{cases} \tag{5}$$

Tire model parameters were selected by performing a ramp steer maneuver, in which the vehicle traveled at a constant u_x and δ was increased at a slow, constant rate. The resulting model and measurements from this procedure are shown in Fig. 2. Experimental slip angles were computed with (2) and (3) using measured r, u_x, and u_y. Experimental forces were estimated with the equations of motion (1) using measured r, u_x, and u_y, and a steady state assumption. This forms a system of linear equations which can be solved for F_{yf} and F_{yr}. The resulting model, shown in Fig. 2, represents the lumped effect of the right and left tires and is a good fit for the experimental data.

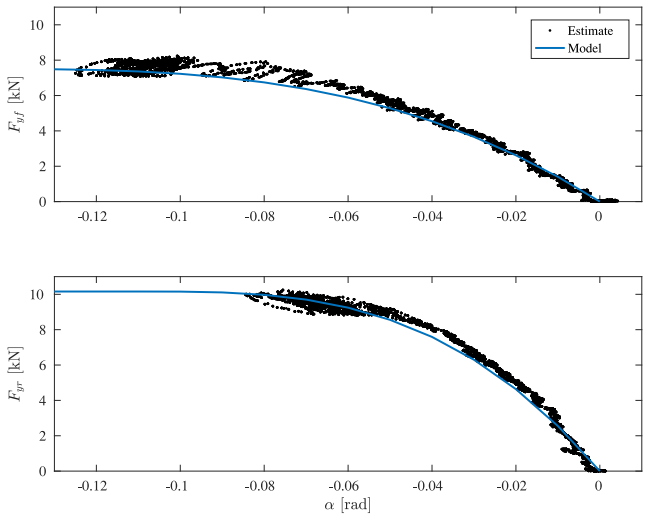


Fig. 2. Modeled lateral tire forces and estimates from a ramp steer calibration procedure.

D. Weight Transfer

The normal forces on the front and rear tires are modeled by assuming a static weight distribution and by adding the effect of steady-state longitudinal weight transfer under small steering angles:

$$F_{zf} = \frac{1}{L}(mbg - h_{cm}(F_{xf} + F_{xr})) \tag{6}$$

$$F_{zr} = \frac{1}{L}(mag + h_{cm}(F_{xf} + F_{xr})) \tag{7}$$

where h_{cm} is the height of the center of mass above the ground. This model for longitudinal weight transfer is equivalent to assuming a stiff suspension. While a real vehicle will have a compliant suspension, this weight transfer model does not introduce any additional states and is accurate in steady-state. Critically, it captures the effect of increased F_{zf} when braking, which increases the capability for both lateral and longitudinal force on the front axle. Additionally, braking unloads the rear axle, decreasing F_{zr} and the lateral and longitudinal force capability on the rear axle. Lateral weight transfer is not explicitly modeled, but its steady-state effect is lumped into the tire model in the single-track approximation.

The resulting vehicle model provides a complete mapping from states $x = [r, u_y, u_x, \Delta\psi, s, e]$ and inputs $u = [\delta, F_{xf}, F_{xr}]$ to state derivatives:

$$\dot{x} = f_{\text{cont}}(x, u) \tag{8}$$

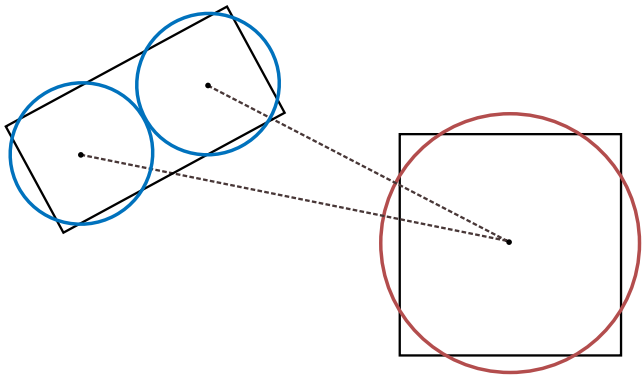


Fig. 3. Representation of the vehicle with two circles (blue) and an obstacle with one circle (red). Collision and distance to an obstacle can be approximated using the signed distance between each vehicle-obstacle circle pair.

III. Obstacles

A fundamental task of this controller is obstacle avoidance, so a good representation of the vehicle and obstacles is very important. In the planning literature, it is common to treat the vehicle or robot as a point and plan in the configuration space [17]. With this approach, obstacles are enlarged and lifted in higher dimensions to account for rotation of the robot, multiple joints, etc. This works well for sampling based methods where obstacle collision can be treated like a black box and queried. However, the lack of an analytical expression for distance or signed distance to obstacles makes planning in the configuration space challenging for a numerical optimization scheme.

Another approach is to treat the vehicle as a point in position space and to enlarge the obstacles to account for the vehicle's size. To account for the rotation of the vehicle, obstacles can be enlarged further [8] or can be modified according to a linearization of the configuration space obstacle [18]. While these approximations are more conducive to numerical optimization, they tend to overly enlarge the obstacle to remain conservative.

Encoding both the vehicle and the obstacle as full-dimensional is more difficult, but enables more precise handling of rotation. Perhaps the most straightforward approach is to numerically linearize the signed distance between the robot and obstacle, which may require special consideration when the true signed distance is not differentiable [11], [12]. Another approach is to assume polytopic or elliptic representations of the robot and invoke Farkas' lemma to replace the obstacle avoidance constraint with its dual equivalent [13], [14], [19]. This constraint is smooth and amenable to numerical optimization at the cost of additional variables and constraints, which can greatly increase problem size and solve times.

The representation of the vehicle and obstacles needs to precisely account for large rotations of the vehicle while remaining simple enough for real-time control. To accomplish this, we represent the vehicle with a set of N_v circles. Each vehicle circle V_i (for $i = 1 \dots N_v$) is parameterized by its radius and center, which depend on the vehicle's position and heading. Similarly, we can represent any obstacles with a set of N_o circles, O_j for $j = 1 \dots N_o$. Fig. 3 shows a vehicle represented with two circles and an obstacle represented with one. This representation enables fast computation of the signed distance d between any

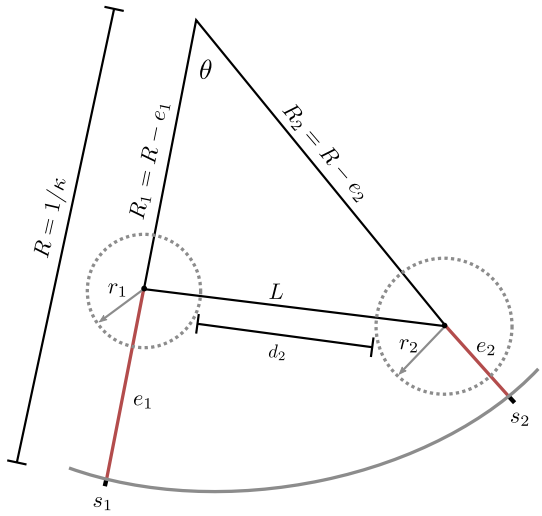


Fig. 4. Signed distance between two circles in a Cartesian frame parameterized by their curvilinear coordinates.

vehicle circle V_i and obstacle circle O_j. If all of the signed distances between all vehicle-obstacle circle pairs are positive, then the vehicle and obstacle representations do not intersect and there is no modeled collision.

For two arbitrary circles C_1 and C_2, each parameterized by their position in curvilinear coordinates and radius (s_1, e_1, r_1) and (s_2, e_2, r_2), respectively, the signed distance in the curvilinear coordinate system is:

$$d_1(C_1, C_2) = \sqrt{(s_2 - s_1)^2 + (e_2 - e_1)^2} - r_2 - r_1 \tag{9}$$

When this distance is positive, it represents the distance separating C_1 and C_2. When it is negative, it represents the distance of penetration, or amount of overlap, between C_1 and C_2.

The signed distance between two circles is easy to compute in the curvilinear space, but distances are not the same as in a Cartesian space. To analyze the approximation error of (9) due to curvature, we compare it to the signed distance in a Cartesian space, as shown in Fig. 4. Under the assumption that the two circles are sufficiently close that the curvature of the reference line κ is approximately constant between them, the angle θ is $\kappa(s_2 - s_1)$. The distance between the center of the circles L can be found using the law of cosines:

$$\begin{aligned} L^2 &= R_1^2 + R_2^2 - 2R_1R_2\cos\theta \\ &= \left(2R^2 - 2Re_1 - 2Re_2\right)(1 - \cos\theta) \\ &\quad + e_1^2 + e_2^2 - 2e_1e_2\cos\theta \end{aligned} \tag{10}$$

The assumption that $\theta = \kappa(s_2 - s_1)$ is small, which is true for typical road curvatures and when circles are relatively close, enables the approximation $\cos\theta \approx 1 - \theta^2/2$:

$$\begin{aligned} L^2 &= \left(2R^2 - 2Re_1 - 2Re_2\right)\left(\frac{\kappa^2(s_2 - s_1)^2}{2}\right) \\ &\quad + e_1^2 + e_2^2 - 2e_1e_2\left(1 - \frac{\kappa^2(s_2 - s_1)^2}{2}\right) \\ &= (1 - e_1\kappa)(1 - e_2\kappa)(s_2 - s_1)^2 + (e_2 - e_1)^2 \end{aligned} \tag{11}$$

The signed distance in a Cartesian frame, under the above assumptions, is therefore:

$$d_2 = \sqrt{(1-e_1\kappa)(1-e_2\kappa)(s_2-s_1)^2 + (e_2-e_1)^2} - r_2 - r_1 \tag{12}$$

This is identical to the intuitive distance d_1 when either the curvature is zero or the two circles lie on the reference line, and varies smoothly as those parameters change. While d_1 can be used to represent the signed distance, it will differ from the Cartesian distance d_2 by the above $(1-e_1\kappa)(1-e_2\kappa)$ factor, overestimating the distance when $e_1\kappa > 0$ and $e_2\kappa > 0$. Although the curvature of roads is typically very small relative to the width of the road, d_2 more accurately models distance, especially on high curvature roads.

IV. Problem Formulation

The controller repeatedly solves an optimal control problem to compute an input trajectory of δ, F_{xf}, and F_{xr} to safely guide the vehicle through the scenario. It does not attempt to solve the combinatorial problem of deciding which side each obstacle should be passed and in what order or if the vehicle should come to a stop before certain obstacles. Instead, the controller assumes that a higher level planner has selected a driving corridor, indicated by desired trajectories for lateral position and speed. We stress, however, that these desired trajectories can be very crude and are used to indicate to the controller which general homotopic class of solutions it should explore. It is the responsibility of the controller to take this rough decision and compute a full state trajectory that is safe and dynamically feasible.

A. Discretization

The optimal control problem is transformed to a nonlinear program by a simultaneous transcription method. This method considers the values of the states and inputs at discrete points in time and transforms the differential equation of the model in (8) to a difference equation using a numerical integration scheme, assuming piece-wise constant inputs between stages.

It is important to choose an integrator with enough complexity to accurately integrate the equations of motion but one that is also simple enough to run quickly. A very common scheme is Euler's method, or forward Euler:

$$x(t+h) = x(t) + hf(x(t), u(t)) \tag{13}$$

Another method is an explicit second order Runge-Kutta integrator (RK2) sometimes referred to as the midpoint method:

$$\begin{aligned} k_1 &= hf(x(t), u(t)) \\ k_2 &= hf\left(x(t) + \frac{k_1}{2}, u(t)\right) \\ x(t+h) &= x(t) + k_2 \end{aligned} \tag{14}$$

A given integrator's performance depends on both the amount of time between stages h and the dynamics model. While bounds on error can be computed (the total accumulated error of Runge-Kutta 4 is famously $O(h^4)$), it is difficult to exploit prior knowledge of the differential equations to prescribe an acceptable fixed step size.

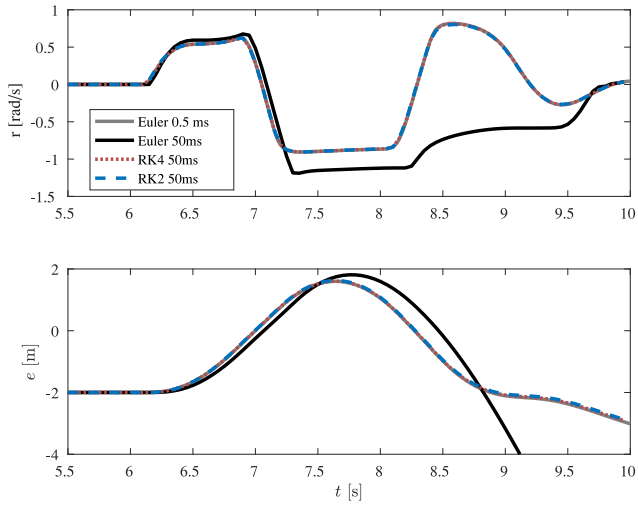


Fig. 5. Results from integrating the given vehicle model and a reasonable trace of inputs with candidate integrators. Runge-Kutta 2 provides a good balance between simplicity and accuracy.

Fig. 5 shows the results of integrating a trace of inputs (δ, F_{xf}, F_{xr}) and the given vehicle model (8) with different integrators for the vehicle states r and e. The input trace comes from an emergency double lane change and represents an intense maneuver in that the state derivatives have high magnitude and the state is in the saturated region of the tire model for a significant amount of time. Euler's method with a step size of $h = 0.5$ ms is presented as a baseline. Euler's method with a step size of $h = 50$ ms is very inaccurate, whereas for the same step size, RK2 performs very well. Furthermore, RK2 performs just as well as RK4 for this application with significantly less computation. For these reasons, we selected RK2 as the discretization method.

B. Change of Variables

It may be natural to use F_{xf} and F_{xr} as the longitudinal input variables in the optimization problem, but there may be a better choice given the application. It is important to minimize or limit longitudinal jerk, so the problem needs to include $\dot{F}_{xf}$ and $\dot{F}_{xr}$ as variables in addition to F_{xf} and F_{xr}, for a total of four longitudinal variables per stage. Additionally, we want to constrain F_{xf} and F_{xr} to have the same sign to prevent the drive motor and brakes from competing with each other. A natural encoding of this constraint is:

$$F_{xf}F_{xr} \geq 0 \tag{15}$$

However, this constraint has a saddle point at $(F_{xf} = 0,\ F_{xr} = 0)$ which is a typical location in the input space. Because the Hessian of this constraint is not positive definite, it can contribute to the Hessian of the Lagrangian of the optimization problem not being positive definite. This undermines the Newton

step at the core of the solver, which requires a positive definite Hessian. Including this constraint is detrimental to the solver's speed and robustness.

An alternative choice of variables is total longitudinal force F_x and brake distribution (or bias) λ_{brake}. The brake distribution represents what fraction of the total longitudinal force should be applied on the front axle; the remaining force should be applied on the rear axle. Given F_x, λ_{brake}, and a fixed drive distribution λ_{drive}, F_{xf} and F_{xr} can be computed:

$$F_{xf} = \begin{cases} \lambda_{\text{brake}} F_x & F_x \leq 0 \\ \lambda_{\text{drive}} F_x & F_x > 0 \end{cases} \tag{16}$$

$$F_{xr} = \begin{cases} (1-\lambda_{\text{brake}}) F_x & F_x \leq 0 \\ (1-\lambda_{\text{drive}}) F_x & F_x > 0 \end{cases} \tag{17}$$

This choice of variables implicitly encodes the constraint that F_{xf} and F_{xr} have the same sign. Additionally, the controller can minimize longitudinal jerk by including $\dot{F}_x$ as an optimization variable. Because the brake system can direct force between the front and rear axle much faster than the other dynamics in the system, we do not need to include $\dot{\lambda}_{\text{brake}}$. Under this assumption, the problem needs only three longitudinal variables per stage: $\dot{F}_x$, F_x, and λ_{brake}, one less than the formulation with F_{xf} and F_{xr}.

C. Cost Function

The objective of the controller is to roughly track some sequence of desired lateral positions and longitudinal speeds while finding a feasible state trajectory that avoids becoming too close to or colliding with obstacles or road edges. The cost function is:

$$\begin{aligned} J = \sum_{k=1}^{N} \Bigg(& Q_e (e^k - e^k_{\text{des}})^2 + Q_{ux} (u^k_x - u^k_{x,\text{des}})^2 \\ & + Q_\delta (\dot{\delta}^k)^2 + Q_{Fx} (\dot{F}^k_x)^2 + Q_\lambda (\lambda^k_{\text{brake}} - \lambda_{\text{nat}})^2 \\ & + \sum_{i=1}^{N_v} \sum_{j=1}^{N_o} p_{\text{obs}}(d_2(V^k_i, O^k_j)) \\ & + \sum_{i=1}^{N_v} p_{\text{edge}}(d_2(V^k_i, r_{\text{left}}(s^k))) \\ & + p_{\text{edge}}(d_2(V^k_i, r_{\text{right}}(s^k))) \Bigg) \end{aligned} \tag{18}$$

where Q_e and Q_{ux} are quadratic weights on lateral position error and longitudinal speed error, Q_δ and Q_{Fx} are quadratic weights on $\dot{\delta}$ and $\dot{F}_x$, and Q_λ is a small quadratic weight that pushes λ_{brake} towards a typical value denoted by λ_{nat}. The continuous and differentiable penalty functions p_{obs} and p_{edge} are zero above a certain distance and quadratic below it.

$$p_\star = \begin{cases} 0, & d > d_{\text{min},\star} \\ Q_{\text{dist}} (d - d_{\text{min},\star})^2, & d < d_{\text{min},\star} \end{cases} \tag{19}$$

for $\star = [\text{obs,edge}]$.

This means a trajectory only incurs cost from p_{obs} or p_{edge} if it causes the vehicle to come within $d_{\text{min,obs}}$ of an obstacle or within $d_{\text{min,edge}}$ of a road edge, indicated by $r_{\text{left}}(s)$ and $r_{\text{right}}(s)$. Q_{dist} is chosen to dominate the cost function if this occurs; the controller will accept lateral positions and longitudinal speeds that differ from the desired values if they allow for a collision free trajectory.

The state of the switch in (19) is not known *a priori*, but instead can change mid-optimization as the solver moves the planned vehicle states nearer to or further from obstacles as needed. An implicit function of the controller is to determine when the vehicle must come close to some obstacles to avoid outright collision with others, so this information is not constrained ahead of time. Although the penalty function is defined piecewise, it is continuous and has continuous first derivatives with respect to the optimization variables, making it amenable to numerical optimization.

Increasing the amount of circles used (N_v, N_o) enables designers to more accurately capture the shape of the vehicle or obstacles, for example, placing smaller circles at the corners to create a more rectangular representation. However, these more detailed representations come at a cost. While increasing N_v and N_o does not increase the number of optimization variables or constraints, the computational cost to evaluate (18) increases as $O(NN_vN_o)$. The results presented in this paper use $N_v = 2$, the minimum number to capture vehicle rotation, and $N_o = 2$, one for each obstacle.

D. Nonlinear Program and Solver

Whereas previously we have used u to represent inputs to the model, we now define $\bar{u}$ to represent the highest order derivatives of u used in the cost function, and we define $\bar{x}$ to represent the state x augmented with δ and F_x. For this problem, $\bar{u} = [\dot{\delta}, \dot{F}_x, \lambda_{\text{brake}}]$ and $\bar{x} = [\delta, F_x, r, u_y, u_x, \Delta\psi, s, e]$. In terms of these variables, the nonlinear problem is:

$$\begin{aligned} \min \quad & J \\ \text{s.t.} \quad & \bar{x}^1 = \bar{x}_{\text{meas}} \\ & \lambda^1_{\text{brake}} = \lambda_{\text{brake,meas}} \\ & \bar{x}^{k+1} = f_{\text{dis}}(\bar{x}^k, \bar{u}^k) \qquad k = 1, \ldots, N-1 \\ & \bar{u}_{\text{LB}} \leq \bar{u}^k \leq \bar{u}_{\text{UB}} \qquad k = 1, \ldots, N \\ & \delta_{\text{LB}} \leq \delta^k \leq \delta_{\text{UB}} \qquad \vdots \\ & F^k_x \leq F_{x,\text{max}} \\ & |F^k_{xf}| \leq \gamma \mu F^k_{zf} \\ & |F^k_{xr}| \leq \gamma \mu F^k_{zr} \end{aligned} \tag{20}$$

where f_{dis} is the vehicle model (8) and the first-order relation between (δ, F_x) and $(\dot{\delta}, \dot{F}_x)$ discretized with a Runge-Kutta 2 integrator (14). There are upper and lower bounds on $\bar{u}$ and δ and upper bounds on F_x that enforce actuator limits.

The amount of force available at each tire is not constant but varies as the normal load changes. The total amount of force

TABLE I
CONTROLLER PARAMETERS

Parameter	Description	Value
N	number of stages in prediction horizon	50
h_{plan}	number of ms between stages	50
h_{replan}	number of ms before replanning	50
Q_e	weight for lateral position error	$\frac{1}{(0.50)^2}$
Q_{ux}	weight for longitudinal speed error	$\frac{1}{(5)^2}$
Q_δ	weight for steering rate	$\frac{1}{(10\cdot\pi/180)^2}$
Q_{Fx}	weight for longitudinal force rate	$\frac{1}{(10e3)^2}$
Q_λ	weight for brake distribution	0.1
λ_{nat}	"natural" brake distribution	0.7
Q_{dist}	weight for signed distance to obstacles	$\frac{1}{(0.10)^2}$
$d_{\text{min,obs}}$	minimum safe distance to obstacles in m	0.7
$d_{\text{min,edge}}$	minimum safe distance to road edges in m	0.5

a tire can produce is μF_z: this is the friction circle constraint. The lateral tire model enforces this constraint to limit the lateral forces given the longitudinal forces, but the problem must also constrain the longitudinal forces. Both F_{xf} and F_{xr}, computed using (17), are limited by μF_{zf} and μF_{zr}, computed using (7).

An additional factor of γ controls what fraction of modeled total force is available for longitudinal force. If $\gamma = 1$, F_{xf} or F_{xr} can use all available force, leaving $F_{yf} = 0$ or $F_{yr} = 0$. Due to the limitations in precisely estimating friction, it can be useful to limit F_{xf} and F_{xr} with a slightly lower value of γ, making the controller optimistic with respect to maximum lateral force. The experiments presented in this paper were conducted with $\gamma = 0.95$.

The optimization problem is solved in terms of the variables δ, F_x, and λ_{brake} and converted to δ, F_{xf}, and F_{xr} using (17). The latter variables are then commanded to lower level controllers as targets to track. The cost function weights and other parameters of the problem are shown in Table I.

The problem is solved with the FORCES NLP interior point solver [20], using automatic differentiation of the cost function and constraints from CasADi [21]. The solver approximates the Hessian of the Lagrangian of (20) using BFGS [22] and exploits the sparsity of the problem to improve solve times.

E. Replanning

In comparison to quadratic (or other convex) problems, nonlinear problems require more computation time. In highly dynamic maneuvers, the vehicle state can change significantly during this time, so accounting for computation delays is important. The strategy the controller uses is shown in Fig. 6. Assuming the controller already has a plan, or full state and input trajectory, it does two things simultaneously. First, it executes that plan for the next 50 ms by applying the sequence of inputs δ, F_{xf}, and F_{xr} to the vehicle. The lower level controllers on the test vehicle accept new commands every 10 ms, so every 10 ms the current plan is linearly interpolated to find the correct inputs at that time. This linear interpolation is consistent with the integrator assumption that $\dot{\delta}$, $\dot{F}_x$, and λ_{brake} are constant over a given stage. Second, it simulates the vehicle model with that sequence of inputs to obtain an estimate of what the state will be in 50 ms and begins computing the next plan starting from that simulated initial state.

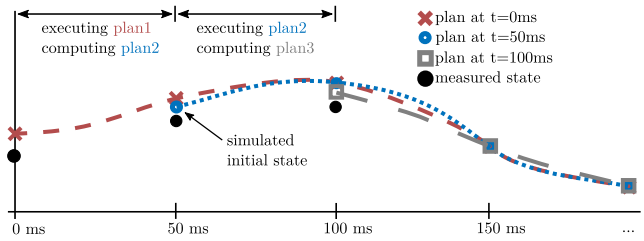


Fig. 6. Prediction horizon shown for three consecutive time steps of a hypothetical scenario. Computation delays are compensated for by simulating the current plan and planning from that predicted initial state while simultaneously executing the current plan.

A planned trajectory is valid for its initial state. Of course, the simulated initial state will never exactly match the measured initial state, but it should be close for an accurate model. This enables the controller to plan trajectories that are valid when computation is finished, not ones that were valid when computation began.

A related strategy is to maintain a full copy of the last computed plan. Even though only the first 50 ms of the planned inputs will typically be used, this copy provides some robustness to computation times. If the solver fails to converge within the time limit of 50 ms, the controller can continue executing commands from the last plan while computing the next plan. This occurred once during the experimental results presented here.

F. Limitations

The objective of the controller is to find a locally optimal trajectory within a homotopic class of solutions, not to decide between homotopies. In other words, the controller depends on a higher level decision maker to select, for instance, whether to pass an obstacle on the right or left. Furthermore within such a selected homotopy, the controller solves a non-convex optimization problem. The interior point method it uses can only find locally optimal solutions, with none of the mathematical guarantees of convergence that can be made for convex problems. Empirically, however, the solver converges in a wide range of conditions and its solutions have been effective in controlling the vehicle in both simulation and experiment.

V. RESULTS

A. Simulation Comparison

The representation of the vehicle and obstacles with circles enables real-time control. To demonstrate this, we compare the proposed controller to one that one that is identical except for a rectangular vehicle and obstacle representation. This representation is exact, with no approximation, but requires an additional variable per side of the vehicle and each obstacle. For three rectangles, these 12 variables per stage are used in two additional scalar inequality constraints and two scalar equality constraints per stage to enforce a collision free path, as described by Zhang [14].

In simulation, the vehicle drove at $u_{x,\text{des}} = 14$ m/s in the right lane on a straight road. One obstacle blocked the right lane and a second obstacle 15 m further along the road blocked the left

TABLE II
SOLVE TIMES

Method	Mean (ms)	Median (ms)	Max (ms)
Rectangular (sim)	136	67	726
Circular (sim)	15	12	70
Circular (exp)	28	27	60

Fig. 7. X1: a versatile test vehicle used in these experiments.

TABLE III
VEHICLE MODEL PARAMETERS

Parameter	Description	Unit	Value
a	distance between CoM and front axle	m	1.53
b	distance between CoM and rear axle	m	1.23
m	mass	kg	2000
I_{zz}	moment of inertia in vertical direction	kg·m^2	3764
h_{cm}	height of CoM above ground	m	0.3
$C_{\alpha f}$	front cornering stiffness	kN/rad	150
$C_{\alpha r}$	rear cornering stiffness	kN/rad	280
μ	coefficient of friction	-	0.9
C_{d0}	constant drag term	N	241
C_{d1}	linear drag coefficient	N·s/m	25.1
$\delta_{\text{UB/LB}}$	bounds on steering angle	deg	± 18
$\dot{\delta}_{\text{UB/LB}}$	bounds on steering angle rate	deg/s	± 90
$F_{x,\max}$	maximum force due to motor torque limit	kN	7.2

lane. A proportional depiction of the circular and rectangular representations used in the simulations is shown in Fig. 3. Both controllers successfully guided the vehicle through a double lane change to avoid the obstacles. The two controllers produced similar control inputs with different solve times, which are shown in Table II.

The rectangular (and more generally, polytopic) representation allows for very precise handling of the vehicle and obstacles, but requires too much computation time to run in real time for this application. The circular representation is an acceptable approximation that is simple enough for real-time control.

B. Experimental Platform

Experiments were performed using X1, a student-built electric vehicle. Model parameters for X1 are shown in Table III. An onboard NovAtel SPAN-SE navigation system fused Global Navigation Satellite System (GNSS) measurements with inertial measurements from gyroscopes and accelerometers to provide accurate position, velocity, and acceleration estimates of the center of mass, as well as attitude and angular velocity of the vehicle.

The controller was run on a conventional onboard computer running Ubuntu 16.04 with an i7-6700 CPU running at 4.0 GHz. Commands of steering angle, braking force on each axle (mapped to pressure), and drive force (mapped to torque) were passed to a hard real-time computer for lower-level control. Cost function weights, as shown in Table I, were chosen such that reasonable values for each term were scaled to unity. For example, a lateral position error of 50 cm, a steering rate of 10 deg/s, and being positioned 10 cm inside an obstacle's minimum safe distance all contribute a value of 1 to the total cost.

C. Experiment

In this experiment, the test vehicle drove on a straight road, staying in the middle of its lane. After the center of mass of the vehicle passed the $s = 180$ line (Time ① in Fig. 8), two obstacles appeared: one in the vehicle's lane at $s = 200$ and one in the oncoming lane at $s = 215$. At this point, the vehicle began an emergency double lane change. Times ①–⑤ are represented by vertical gray dashed lines in Figs. 9, 10, 11, and 12. Fig. 13 shows a composite top view of the vehicle at approximately Times ①–⑤ for additional context.

The closed-loop inputs computed by the controller are shown in Fig. 9. As soon as the obstacles appeared at Time ①, the controller began steering left, limited by the slew rate, until it reached the vehicle's maximum steering angle of 18°. Simultaneously, the controller immediately reduced F_{xr} to -4 kN, a moderate braking force. The effect of this is twofold. First, braking slowed the vehicle, making the maneuver easier by allowing more time to accelerate the mass of the vehicle laterally to avoid the obstacles. Additionally, braking transferred weight to the front of the vehicle, which increases the lateral force capability of the front axle. This additional force capability was utilized by the controller; Figure 10 (top) shows both the lateral force and its limit increased between Time ① and Time ②.

Conversely, the rear lateral force shown in Fig. 10 had a reduced capacity after Time ①, both due to weight transferring off the rear axle and the derated maximum lateral force due to the applied longitudinal force (4). When the obstacles appeared at Time ①, all available force on the front axle was needed for turning, but the rear axle was free to brake until the vehicle had rotated enough to require rear lateral force. The controller was able use all available frictional force on the front axle to turn the vehicle while still braking the rear axle for the few hundred milliseconds in which it was otherwise not needed. At Time ②, the controller began braking the front axle (Fig. 9 middle) and easing off the brakes on the rear axle, increasing F_{xr} to zero (Fig. 9 bottom). The front axle was saturated by Time ②, and the increased front braking force implied reduced front lateral force, shown in Fig. 10 (top).

By Time ③, F_{xf} was close to its minimum value, leaving a only a small amount of force left for F_{yf}. By this time, the vehicle had rotated enough to avoid the immediate danger of the first obstacle, allowing the controller to allocate time braking on

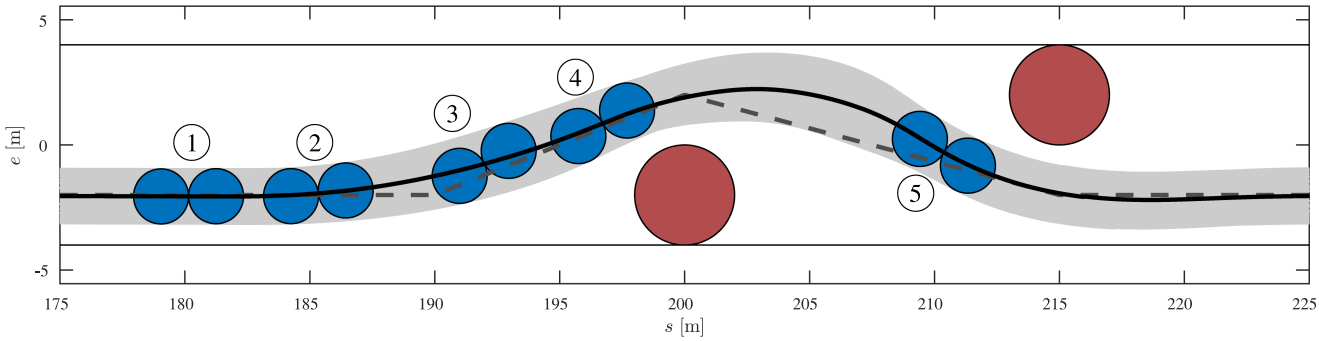


Fig. 8. A top-down view of the pop-up double lane change experiment. The trajectory of the vehicle's CoM is shown in black, rough desired lateral position is shown with a dashed line, vehicle positions at five particular points in time are shown in blue, vehicle positions throughout the experiment are shown in light gray, and obstacles are shown in red. At Time ①, both obstacles appeared. At Time ②, the rear tires became saturated. At Time ③, the vehicle reached maximum brake force on the front tires. At Time ④, the vehicle reached maximum brake force on the rear tires. At Time ⑤, the rear tires began to leave the saturated region.

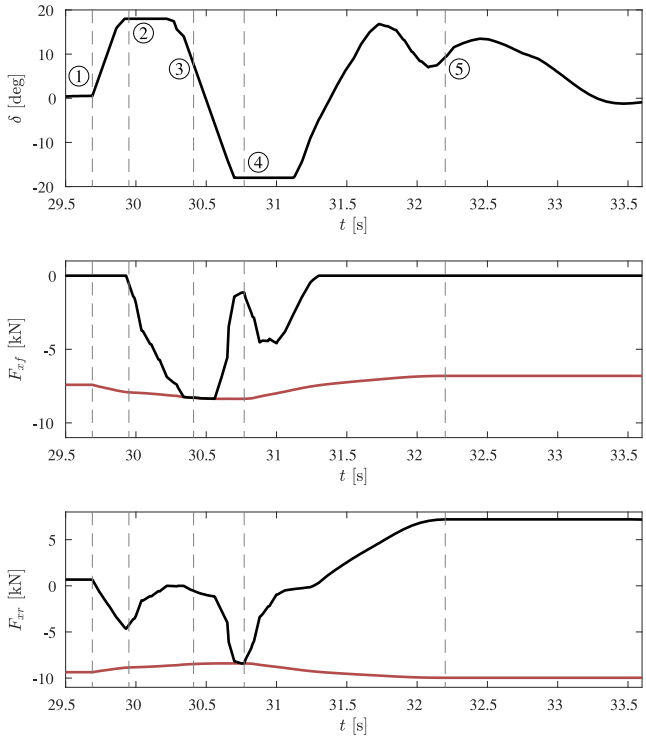


Fig. 9. Steering angle δ (top), front longitudinal force F_{xf} (middle), and rear longitudinal force F_{xr} (bottom) as commanded by the controller. Modeled force limits are shown in red.

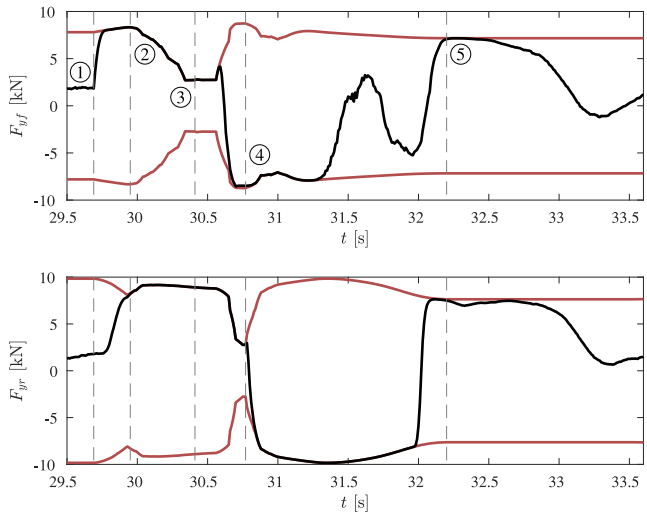


Fig. 10. Lateral forces (black) and their limits (red) for the front (top) and rear (bottom) tires, both as modeled by the controller.

the front axle instead of steering. Because the rear axle needed a significant amount of lateral force to stabilize and rotate the vehicle at approximately Time ③, the controller was unable to brake the rear axle. By Time ④, the situation was reversed: the vehicle needed to steer to the right and produce large negative force to avoid going off the road, so it reduced the front braking force to almost zero. Simultaneously, there was a brief period of time where the necessary stabilizing rear lateral tire force transitioned from positive to negative; the controller exploited this brief time to brake the rear axle.

At Time ④, the vehicle was fairly close to the first obstacle. The minimum distance between every vehicle circle and every obstacle circle, and between every vehicle circle and the road edge, is shown in Fig. 11. The controller needed to steer enough to avoid going off the road or hitting the second obstacle but not so much as to drive too close to the first obstacle. A similar trade-off is made just before Time ⑤, where the controller reduced the steering angle (Fig. 9 top), resulting in a negative F_{yf} that pushed the vehicle away from the second obstacle, but not so much so that the vehicle could not stay on the road.

It is clear that the controller made large trade-offs between lateral and longitudinal forces, slowing the vehicle when possible and necessary to complete the double lane change. The longitudinal speed through the maneuver is shown in Fig. 12. The controller was able to slow the vehicle from around 17.5 m/s to around 8.5 m/s in a little over 2 s. Despite this large reduction in speed, both the front and rear axles saturated near Time ⑤ as the vehicle passed the second obstacle. The vehicle would not have been able to complete the maneuver without hitting either

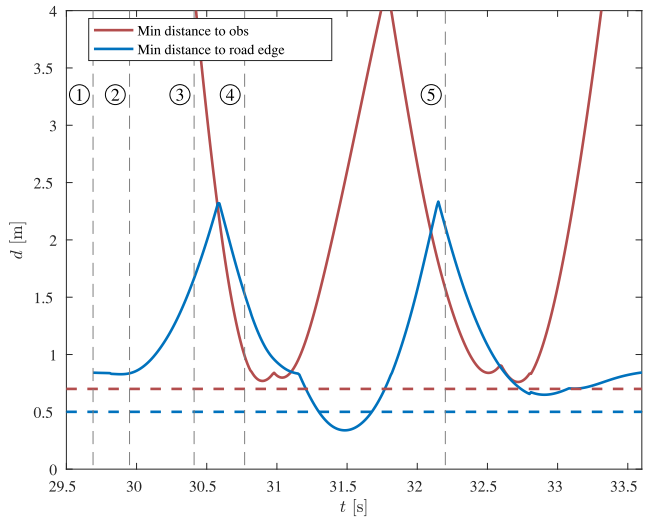


Fig. 11. Minimum signed distance between every vehicle circle and every obstacle circle, and between every vehicle circle and the road edge. The minimum distance before cost is incurred, $d_{\text{min,obs}}$ and $d_{\text{min,edge}}$ is shown with dashed lines.

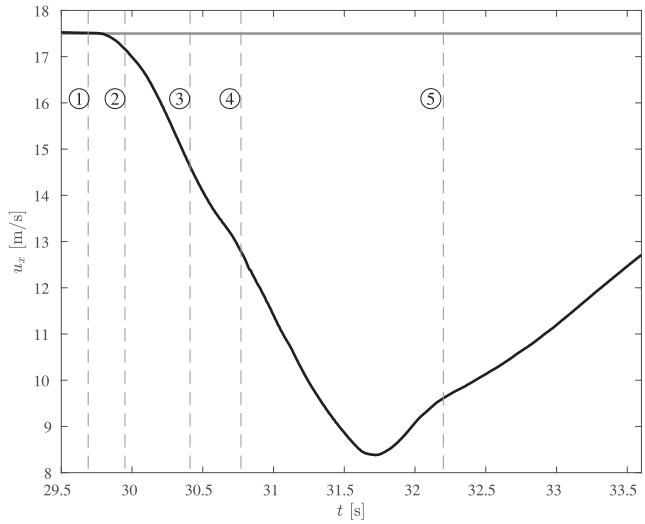


Fig. 12. Longitudinal speed u_x (black) and desired speed (gray) throughout the maneuver. The controller was able to reduce speed from over 17.5 m/s to under 8.5 m/s, while avoiding the obstacles. As soon as it was safe to do so, it began accelerating back to the target speed.

obstacle or going off the road if it was traveling at the initial speed of 17.5 m/s. Furthermore, because the controller relied so heavily on distributing braking forces between the front and rear axles, it would be difficult or potentially impossible for a human driver using a single brake pedal to complete this maneuver without a collision.

VI. Conclusion

In this paper, we present a nonlinear model predictive controller capable of producing steer angles and front and rear longitudinal forces to safely guide a vehicle through potentially dangerous situations. The vehicle model, integrator, and representation of the vehicle and obstacles are chosen to enable fast computation and therefore a fast reaction to changes in the environment. The representation of the vehicle and obstacles is sufficiently detailed to capture effects of vehicle rotation for potential collisions, and enables the controller to reason about rotation and longitudinal and lateral position. The vehicle and tire models capture the dynamics of the vehicle and, critically, inform the controller of the coupling between lateral and longitudinal forces. Experimental results of a test vehicle performing an emergency double lane change demonstrate the ability of the controller to balance lateral and longitudinal forces on each axle to avoid collisions.

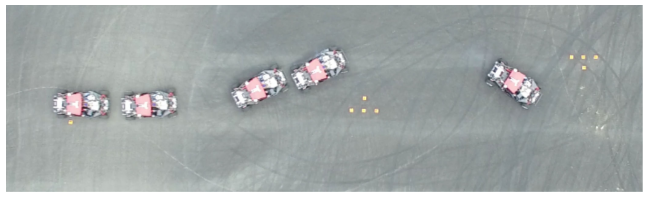


Fig. 13. Top view of experiment with vehicle shown at approximately Times ①–⑤.

Acknowledgment

The authors would like to thank Larry Cathey for his support running experiments and maintaining X1, Embotech for providing FORCES Pro Academic Licenses, and the students of the DDL for thoughtful discussions about modeling and optimization.

References

[1] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat, "Predictive active steering control for autonomous vehicle systems," *IEEE Trans. Control Syst. Technol.*, vol. 15, no. 3, pp. 566–580, May 2007.

[2] P. Falcone, F. Borrelli, H. E. Tseng, J. Asgari, and D. Hrovat, "Linear time-varying model predictive control and its application to active steering systems: Stability analysis and experimental validation," *Int. J. Robust Nonlinear Control*, no. July 2007, pp. 862–875, 2007, doi: 10.1002/rnc.1245.

[3] M. Brown, J. Funke, S. Erlien, and J. C. Gerdes, "Safe driving envelopes for path tracking in autonomous vehicles," *Control Eng. Pract.*, vol. 61, pp. 307–316, 2017, doi: 10.1016/j.conengprac.2016.04.013.

[4] J. Funke, M. Brown, S. M. Erlien, and J. C. Gerdes, "Collision avoidance and stabilization for autonomous vehicles in emergency scenarios," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 4, pp. 1204–1216, Jul. 2017.

[5] C. E. Beal and J. C. Gerdes, "Model predictive control for vehicle stabilization at the limits of handling," *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 4, pp. 1258–1269, Jul. 2013.

[6] P. Falcone, H. E. Tseng, F. Borrelli, J. Asgari, and D. Hrovat, "MPC-based yaw and lateral stabilisation via active front steering and braking," *Vehicle Syst. Dyn.*, vol. 46, no. SUPPL.1, pp. 611–628, 2008.

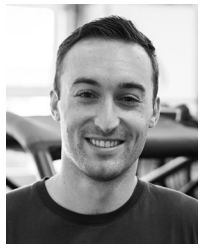
[7] J. Liu, P. Jayakumar, J. Stein, and T. Ersal, "Combined speed and steering control in high speed autonomous ground vehicles for obstacle avoidance using model predictive control," *IEEE Trans. Veh. Technol.*, vol. 66, no. 10, pp. 8746–8763, Oct. 2017, doi: 10.1109/TVT.2017.2707076.

[8] H. Febbo, J. Liu, P. Jayakumar, J. L. Stein, and T. Ersal, "Moving obstacle avoidance for large, high-speed autonomous ground vehicles," in *Proc. Amer. Control Conf.*, 2017, pp. 5568–5573.

[9] Y. Gao, T. Lin, and E. Tseng, "Predictive control of autonomous ground vehicles with obstacle avoidance on slippery roads," in *PRoc. ASME Dyn. Syst. Control Conf.*, 2010, vol. 1, pp. 265–272.

[10] J. V. Frasch *et al.*, "An Auto-generated Nonlinear MPC Algorithm for Real-Time Obstacle Avoidance of Ground Vehicles," in *Proc. Eur. Control Conf.*, 2013, pp. 4136–4141.

[11] J. Schulman *et al.*, "Motion planning with sequential convex optimization and convex collision checking," *Int. J. Robot. Res.*, vol. 33, no. 9, pp. 1251–1270, 2014.
[12] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for Bertha - A local, continuous method," in *Proc. IEEE Intell. Vehicles Symp.*, 2014, pp. 450–457.
[13] C. Landry, D. Hömberg, R. Henrion, and M. Gerdts, "Path planning and collision avoidance for robots," *Numer. Algebra, Control Optim.*, vol. 2, no. 3, pp. 437–463, 2012. [Online]. Available: http://www.aimsciences.org/journals/displayArticlesnew.jsp?paperID=7518
[14] X. Zhang, A. Liniger, and F. Borrelli, "Optimization-based collision avoidance," 2017. [Online]. Available: http://arxiv.org/abs/1711.03449
[15] R. Y. Hindiyeh and J. C. Gerdes, "Design of a dynamic surface controller for vehicle sideslip angle during autonomous drifting," in *Proc. IFAC Proc. Volumes*, 2010, pp. 560–565.
[16] H. Pacejka, *Tire and Vehicle Dynamics*. Amsterdam, The Netherlands: Elsevier, 2005.
[17] S. M. LaValle, *Planning Algorithms*. Cambridge U.K.: Cambridge Univ. Press, 2006.
[18] S. Erlien, "Shared vehicle control using safe driving envelopes for obstacle avoidance and stability," Ph.D. dissertation, Stanford University, 2015.
[19] C. Liu, S. Lee, S. Varnhagen, and H. E. Tseng, "Path planning for autonomous vehicles using model predictive control," in *Proc. IEEE Intell. Vehicles Symp.*, 2017, pp. 174–179.
[20] A. Zanelli, A. Domahidi, J. Jerez, and M. Morari, "FORCES NLP: An efficient implementation of interior-point methods for multistage nonlinear nonconvex programs," *Int. J. Control*, vol. 93, no. 1, pp. 13–29, 2017.
[21] J. Andersson, J. Åkesson, and M. Diehl, "CasADi: A symbolic package for automatic differentiation andoptimal control," in *Recent Advances in Algorithmic Differentiation*, S. Forth, P. Hovland, E. Phipps, J. Utke, and A. Walther, Eds. Berlin, Germany: Springer, 2012, pp. 297–307.
[22] R. Fletcher, *Practical Methods of Optimization*. Hoboken, NJ, USA: Wiley, 2013.



Matthew Brown received the B.S. degree in mechanical engineering from the University of California at Berkeley, Berkeley, CA, USA, in 2013. He is currently pursuing the Ph.D. degree in mechanical engineering with Stanford University, Stanford, CA, USA. His current research interests include architecture of autonomous vehicle systems and the interaction between trajectory planning and following. Mr. Brown was a recipient of the National Science Foundation Graduate Research Fellowship Award in 2013.

J. Christian Gerdes received the Ph.D. degree from the University of California at Berkeley, Berkeley, CA, USA, in 1996.

He is currently a Professor in mechanical engineering with Stanford University, Stanford, CA, USA, and the Director of the Center for Automotive Research at Stanford, Stanford University. He is a Co-Founder of Peloton Technology, Mountain View, CA, USA. His laboratory studies how cars move, how humans drive cars, and how to design future cars that work cooperatively with the driver or drive themselves. When not teaching on campus, he can often be found at the racetrack with students, instrumenting historic race cars, or trying out their latest prototypes for the future.

Prof. Gerdes and his team have been recognized with several awards, including the Presidential Early Career Award for Scientists and Engineers, the Ralph Teetor Award from SAE International, and the Rudolf Kalman Award from the American Society of Mechanical Engineers.