Dynamic Vehicle Drifting With Nonlinear MPC and a Fused Kinematic-Dynamic Bicycle Model

Guillaume Bellegarda and Quan Nguyen, *Member, IEEE*

***Abstract*—In this letter we present a versatile trajectory optimization framework that leverages a fused kinematic-dynamic bicycle model for highly dynamic vehicle drifting maneuvers. Our framework can be used online to generate drifting maneuvers, offline to plan drift parking, and additionally enables online tracking of the offline computed parking maneuvers. Importantly, neither individual kinematic nor dynamic bicycle models alone can be used straightforwardly in an optimization framework to plan nor execute the presented motions, as the former cannot model drifting, and the latter becomes ill-defined at low speeds. We validate our framework in a Gazebo simulation of the MIT RACECAR, where we show several drifting scenarios such as steady-state drifting with a range of desired yaw rates as well as a dynamic drift parking maneuver under noisy conditions, and video results can be found at https://youtu.be/MF1_fS6CQQs.**

***Index Terms*—Nonlinear model predictive control, vehicle drift control, autonomous drifting, autonomous vehicles.**

I. Introduction

CONTROLLING vehicles at their limits of handling has significant implications from both safety and autonomous racing perspectives. For example, in icy conditions, skidding may occur unintentionally, making it desirable to safely control the vehicle back to its nominal working conditions. From a racing perspective, drivers of rally cars drift around turns while maintaining high speeds on loose gravel or dirt tracks.

In this letter, we are specifically interested in dynamic vehicle maneuvers involving both steady-state and transient skidding/drifting. By steady-state, or sustained, drift, we refer to maneuvers such as "donuts", where the vehicle maintains constant speed in a circular motion, while the vehicle's orientation is at an angle to the overall velocity of the body, as shown at left of Figure 1. For transient drift, we refer to maneuvers where this velocity is not constant, such as when performing a dynamic parking maneuver (right of Figure 1), or drifting around a variable turn on a racetrack.

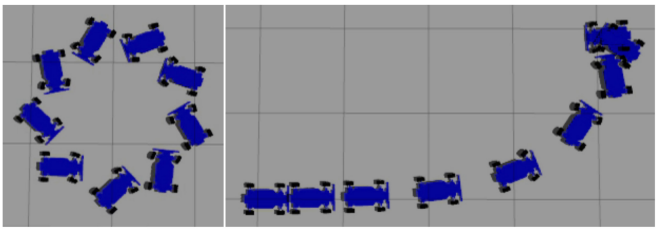

Fig. 1. Dynamic drifting with the MIT RACECAR in Gazebo. **Left:** Steady-state drifting with yaw rate 3 `rad/s`. **Right:** Dynamic drift parking at (4, 2) `m` with a π turn in 2.25 seconds.

A. Related Work

There have been a number of works in these areas involving both sustained and transient drift. Of note are the different vehicle models used as well as the (mostly) decoupling of planning and control with a variety of techniques.

1) Analytical/Optimal: Various analytical and optimal control-based methods have been proposed for vehicle cornering and drifting [1]–[4]. Velenis *et al.* introduced a bicycle model with suspension dynamics and applied numerical optimization to analyze drift cornering behavior [1], [2]. They proposed a framework for achieving maximum corner exit velocity or minimal time cornering, and validated their method in CarSim. Voser *et al.* [3] presented an analytical framework to understand the dynamics of drifting with a simple bicycle model with nonlinear tires, modeling different coefficients of friction between the front and rear tires with the ground. Experimental evidence was collected on a physical all-electric vehicle, P1. Goh and Gerdes [4] presented a controller to drift while tracking a reference path, using a four-wheel model with steady-state weight transfer, with experimental validation on the physical vehicle MARTY.

2) Mixed Open-Loop, Closed-Loop Control: Over short periods of time, even complex dynamics such as vehicle drifting have been shown to be remarkably deterministic [5], leading to mixed open-loop closed-loop (OL-CL) control [5]–[8]. Kolter *et al.* [5] used a probabilistic method called multi-model LQR, which they applied on a real vehicle for a sliding parking maneuver. The vehicle model for normal driving was learned

Manuscript received September 14, 2021; revised November 17, 2021; accepted November 29, 2021. Date of publication December 16, 2021; date of current version December 29, 2021. This work was supported by the USC Viterbi School of Engineering. Recommended by Senior Editor C. Seatzu. *(Corresponding author: Guillaume Bellegarda.)*

Guillaume Bellegarda was with the Department of Aerospace and Mechanical Engineering, University of Southern California, Los Angeles, CA 90007 USA. He is now with the BioRobotics Laboratory, École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland (e-mail: guillaume.bellegarda@epfl.ch).

Quan Nguyen is with the Department of Aerospace and Mechanical Engineering, University of Southern California, Los Angeles, CA 90007 USA (e-mail: quann@usc.edu).

Digital Object Identifier 10.1109/LCSYS.2021.3136142

from experimental data from real driving, and the system was provided with a demonstration of the desired maneuver. The authors of [6]–[8] study drift parking and cornering, and use a rule-based algorithm to plan a reference trajectory tracked with mixed OL-CL. In [8] the authors divide the cornering problem into free, transit, and drifting regions; where path planning is done by a Rapidly Exploring Random Tree (RRT) and rule-based method, with a PI controller for the transit segment.

3) Sampling and Learning-Based: Other works incorporating sampling-based methods for planning include [9], [10]. In [9] the authors investigate using RRT* to generate time optimal trajectories through turns that exploit skidding, and avoid the difficulties in general optimization frameworks of the singularities associated with popular tire models at low speeds. However, the reported computation times are not low enough for real-time implementations. Williams *et al.* [10] present a MPC method called model predictive path integral (MPPI) control, a sampling-based algorithm which can optimize for general cost criteria. The algorithm is validated on a 1/5 scale Auto-Rally vehicle at speeds beyond the friction limits of the vehicle.

Learning-based methods have also been studied in the context of vehicle drifting. Cutler and How [11] achieved autonomous drifting with simulation-aided model-based reinforcement learning on a scaled model car. The throttle is kept constant, and the only input is the steering angle. Lau [12] learned coarse dynamics from a drifting demonstration to guide policy learning for a drifting parking task.

4) Discussion and Limitations: Most of the above works either decouple planning from control by using different models/methods for each, or require demonstrations or learning to achieve desired drifting maneuvers. Ideally, we would want to plan (and then execute) motions through the same well understood real dynamic model.

Additionally, the tire approximations in the above works all rely on some variant of the linear, Fiala, or Pacejka models [13], which all depend on the tire slip angle. This angle becomes singular at low vehicle speeds due to the vehicle velocity term in the denominator, and is thus difficult to use in a general trajectory optimization framework, additionally so because of the discontinuities in the latter two of these models. The methods discussed above attempt to circumvent this issue with a variety of techniques, from sampling-based methods or scenario-specific optimizations, to using different vehicle models for planning and control.

In our previous work, we proposed the linear complementarity problem (LCP) wheel model, and showed that such a model can be used in a nonlinear trajectory optimization framework to generate and control drifting maneuvers for both vehicles [14] as well as a wheel-legged quadruped with passive wheels [15]. More recently, we used MPC based on a kinematic bicycle model as a baseline to ensure a worst-case scenario for deep reinforcement learning, leaving the complex drifting dynamics to be learned for a task of navigating to a series of goal locations as fast as possible [16].

A recently proposed vehicle model interpolates between the kinematic and dynamic bicycle models in hand-tuned velocity ranges in an MPC framework to follow a race track for an autonomous car [17]. For this scenario, drifting is avoided to maintain high speeds around the track.

B. Contribution

In this letter, we show that a fused kinematic-dynamic bicycle model can be used in a general optimization framework to both generate and track dynamic drifting maneuvers. Notably, the model does not have the singularity or discontinuity issues of the commonly used tire approximation models discussed above. Our framework can naturally produce both steady-state drifting at varying desired velocities as well as generate dynamic drift parking maneuvers. We also show that these offline generated trajectories can be tracked online with the same framework to control even transient drifting motions.

Importantly, the dynamic maneuvers generated in this letter would not be possible with either kinematic or dynamic bicycle models alone. At high speeds, the kinematic model does not accurately model the vehicle dynamics, and cannot be used to plan or model any drifting behaviors. Conversely, for motions involving low or 0 velocity in the body x direction (such as when starting from rest, or near the end of parking), the slip angle used by the Pacejka tire model is ill-defined, preventing direct use in trajectory optimization frameworks. With our optimization framework using the fused model, we avoid both of these issues and we are able to successfully plan and control a variety of dynamic drifting maneuvers.

The rest of this letter is organized as follows. Section II describes modeling details for the kinematic, dynamic, and fused kinematic-dynamic bicycle model approximations. Section III describes the nonlinear trajectory optimization framework used to both generate and track motions. Experimental results showing steady-state drifting at various speeds as well as transient drifting for a parking task are presented in Section IV, and a brief conclusion is given in Section V.

II. Modeling

This section describes the vehicle models used in our optimization framework: the kinematic, dynamic, and our fused kinematic-dynamic bicycle model. For further details, the kinematic and dynamic bicycle models are studied for example in [18]. The fused kinematic-dynamic model is inspired from [17], and we adopt similar notation.

A. Dynamic Model

Following from Figure 2, we consider a bicycle model to represent a car with mass m and inertia I_z, where l_R and l_F represent the distance from the center of gravity to the rear and front wheels respectively, $F_{R,y}$ and $F_{F,y}$ are the lateral tire forces of the rear/front wheels, and F_x is the net force in the body direction. The state of the model is written as:

$$\mathbf{x} = [X, Y, \varphi, v_x, v_y, r, \delta]^T \tag{1}$$

which consists of the body position (X, Y) and yaw φ in global coordinates, as well as the longitudinal and lateral velocities (v_x, v_y), the yaw rate r, and steering angle δ. The control inputs

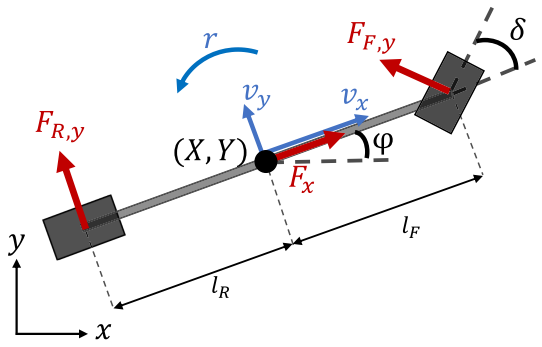

Fig. 2. Dynamic bicycle model, with position vectors in black, velocities in blue, and forces in red.

$\mathbf{u} = [F_x, \Delta\delta]^T$ are the force applied by the wheels in the body direction, and the velocity of the steering angle δ, which is chosen due to its use in the kinematic model. This additionally naturally allows for formulating constraints such as the maximum angular velocity of the steering wheels.

The full equation of motion for the dynamic bicycle model is given by:

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\varphi} \\ \dot{v}_x \\ \dot{v}_y \\ \dot{r} \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} v_x \cos\varphi - v_y \sin\varphi \\ v_x \sin\varphi + v_y \cos\varphi \\ r \\ \frac{1}{m}(F_x - F_{F,y} \sin\delta + m v_y r) \\ \frac{1}{m}(F_{R,y} + F_{F,y} \cos\delta - m v_x r) \\ \frac{1}{I_z}(F_{F,y} l_F \cos\delta - F_{R,y} l_R) \\ \Delta\delta \end{bmatrix} \tag{2}$$

which we succinctly denote as $\dot{\mathbf{x}} = f_{\text{dyn}}(\mathbf{x}, \mathbf{u})$.

The forces $F_{i,y}$ represent the lateral forces between the tires and the ground, where the subscript i refers to the front or rear tires, $i \in \{F, R\}$. We use a simplified Pacejka tire model [13] to represent these forces:

$$\alpha_R = \arctan\left(\frac{l_R r - v_y}{v_x}\right) \tag{3}$$

$$\alpha_F = -\arctan\left(\frac{l_F r + v_y}{v_x}\right) + \delta \tag{4}$$

$$F_{i,y} = D_i \sin\left(C_i \arctan\left(B_i \alpha_i\right)\right) \tag{5}$$

where $\alpha_i, i \in \{F, R\}$ are the front and rear slip angles, and $B_i, C_i, D_i, i \in \{F, R\}$ are experimentally identified coefficients that can be fit from driving the car randomly with variable drifting as in [19].

B. Kinematic Model

The dynamic bicycle model discussed above is ill-defined for low velocities due to v_x appearing in the denominator of the slip angles in Equations (3) and (4). However, slow or zero velocities in the body x direction are important to start and stop vehicle motion, as well as for purely lateral drifting. For slow velocities, kinematic models can accurately model vehicle dynamics, and notably do not depend on the slip angles [18]. We consider the same kinematic model described in [17]:

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\varphi} \\ \dot{v}_x \\ \dot{v}_y \\ \dot{r} \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} v_x \cos\varphi - v_y \sin\varphi \\ v_x \sin\varphi + v_y \cos\varphi \\ r \\ \frac{1}{m} F_x \\ (\dot{\delta} v_x + \delta \dot{v}_x)\frac{l_R}{l_R + l_F} \\ (\dot{\delta} v_x + \delta \dot{v}_x)\frac{1}{l_R + l_F} \\ \Delta\delta \end{bmatrix} \tag{6}$$

which uses the same states as the dynamic model described in Section II-A, and can be summarized as $\dot{\mathbf{x}} = f_{\text{kin}}(\mathbf{x}, \mathbf{u})$.

C. Fused Kinematic-Dynamic Bicycle Model

As the previously discussed kinematic and bicycle models have the same states, it is natural to combine them into one model, where the kinematic model will be used at low speeds, and the dynamic model will be used at higher speeds, for which accurately modeling lateral forces is critical. One straightforward approach is to linearly blend between the two models based on the vehicle speed:

$$\dot{\mathbf{x}} = \lambda f_{\text{dyn}}(\mathbf{x},) + (1 - \lambda) f_{\text{kin}}(\mathbf{x}, \mathbf{u}) = f(\mathbf{x}, \mathbf{u}) \tag{7}$$

In [17], λ is given by a min/max formulation based on the body velocity in the x direction only, v_x. However, such a min/max formulation does not lend itself well to general optimization frameworks. Additionally, for the drifting scenarios we are interested in, the blending will also be inaccurate during purely lateral drifting maneuvers, where the vehicle may have a high v_y component, even if v_x may be low.

To circumvent each of these issues, in this letter we propose representing λ with a sigmoid, specifically, for a desired velocity switching range of $(v_{\text{blend min}}, v_{\text{blend max}})$ as follows:

$$\phi = v_{\text{blend min}} + 0.5(v_{\text{blend max}} - v_{\text{blend min}}) \tag{8}$$

$$\omega = \frac{2\pi}{v_{\text{blend max}} - v_{\text{blend min}}} \tag{9}$$

$$\lambda = \frac{1}{2}\left(\tanh\left(\omega\left((v_x^2 + v_y^2) - \phi\right)\right) + 1\right). \tag{10}$$

This new formulation for λ allows for Equation (7) to be readily used in general optimization frameworks, and since the blending now occurs based on the overall linear velocity of the system, it will be accurate even for purely lateral drifting. Thus, we avoid the singularities of the dynamic model while keeping accuracy at low speeds.

III. Nonlinear Model Predictive Control

This section provides details on formulating the motion planning problem for the system as a nonlinear MPC. At a high level, the full nonlinear system is discretized, and direct collocation along with backward Euler integration is used to generate motion. We first present the general optimization structure, and then explain how the framework can be used for either motion generation or motion tracking.

A. General Framework

The general discrete time optimization can be formulated as follows:

$$\underset{\boldsymbol{x}_k, \boldsymbol{u}_k;\ k=1\cdots N}{\text{minimize}} \quad J(\boldsymbol{x}_N) + \sum_{k=1}^{N} w(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{11}$$

$$\text{subject to} \quad \boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k) \ \text{(Eq. 7)} \tag{12}$$

$$\boldsymbol{x}_0 = \boldsymbol{x}_{\text{init}} \tag{13}$$

$$\boldsymbol{x}_{\min} \leq \boldsymbol{x}_k \leq \boldsymbol{x}_{\max} \tag{14}$$

$$\boldsymbol{u}_{\min} \leq \boldsymbol{u}_k \leq \boldsymbol{u}_{\max} \tag{15}$$

where $\boldsymbol{x}_k$ is the full state of the system at sample k along the trajectory, $\boldsymbol{u}_k$ is the corresponding control input, J and w are final and additive costs that depend on the task (to be explained in Sections III-B and III-C), and N is the total number of samples along the trajectory. Equation (12) constrains the discretized dynamics of the fused kinematic-dynamic model from Equation (7). The initial state of the car is given by the environment, and the states and inputs are bounded by the dynamics as well as a limited (X, Y) range.

B. Motion Generation

While the cost function can be chosen to optimize for various tasks, in this letter we are specifically interested in generating both steady-state and transient drifting, i.e., for a parking task.

1) Steady-State Drifting: For steady-state-drifting, the desired goal is to drift at a particular yaw rate while maintaining the overall body velocity at an offset to v_x. Thus, for a desired yaw rate goal of r_g and body x-component velocity goal v_{x_g}, we can use the following additive cost:

$$w_{ss}(\boldsymbol{x}_k, \boldsymbol{u}_k) = \alpha_{v_x}(v_{x_k} - v_{x_g})^2 + \alpha_r(r_{x_k} - r_{x_g})^2 \tag{16}$$

For steady-state drifting, the final cost J is set to 0. We present results for $\alpha_{v_x} = \alpha_r = 1$, but note that by setting $\alpha_{v_x} = 0$ the MPC will generate natural circular motions that converge to drifting at a particular v_x.

2) Transient Drift Parking: For dynamic parking tasks, we are interested in optimizing to get to a particular goal location for which the vehicle should come to rest at 0 velocity. The parking cost function $J_{park}(\boldsymbol{x}_N)$ is thus defined as the squared error between a set of goal coordinates (i.e., a parking space) (X_g, Y_g, φ_g) and the body coordinates (X_N, Y_N, φ_N), where N is the number of sample points for the trajectory:

$$\begin{aligned} J_{park}(\boldsymbol{x}_N) = {} & (X_g - X_N)^2 + (Y_g - Y_N)^2 + (\varphi_g - \varphi_N)^2 \\ & + (v_{x_N})^2 + (v_{y_N})^2 + (r_N)^2 \end{aligned} \tag{17}$$

where zero velocity at the end of the trajectory is desired. The drift parking optimization is run offline, and we set the additive cost w to 0.

C. Motion Tracking

We use the same formulation to track the trajectories as generated by our framework, such as for the drift parking example described above. More precisely, we would like to track a given reference trajectory of length M, i.e., $(\boldsymbol{x}_{k_{\text{ref}}}, \boldsymbol{u}_{k_{\text{ref}}})$ for $k = 1 \ldots M$. In this case, the horizon of the tracking MPC, N, is shorter than the length of the trajectory of the offline computed drifting maneuver M. For tracking we do not use a final cost function, and we use the following additive cost function:

$$w_{track}(\boldsymbol{x}_k, \boldsymbol{x}_{k_{\text{ref}}}) = (\boldsymbol{x}_k - \boldsymbol{x}_{k_{\text{ref}}})^T \boldsymbol{R} (\boldsymbol{x}_k - \boldsymbol{x}_{k_{\text{ref}}}) \tag{18}$$

where $\boldsymbol{R} = \text{diag}(1, 1, 1, 0.1, 0.1, 0.1, 0.1)$ is a diagonal matrix containing the specified weights. This cost function attempts to minimize the error between the reference and predicted trajectory states. Position errors are more heavily weighted due to the nature of the parking task making accurate position tracking more critical. The variables for the tracking task are initialized with the reference trajectories for the tracking horizon.

D. Implementation Details

The nonlinear MPC is implemented in C++ with CasADi [20], using IPOPT [21] to solve the NLP. We use the MIT RACECAR, which is an All-Wheel Drive (AWD) vehicle, and its publicly available Gazebo simulation, as introduced in [22]. The car has mass $m = 4.78$ kg, inertia $I_z = 0.0665$ kg $\cdot$ m^2, and $l_R = l_F = 0.18$ m. Since the vehicle is AWD, the desired body force F_x from the optimization is converted to torque inputs at each wheel with $\tau_w = F_x w_r / 4$, where w_r is the wheel radius. The control input $\Delta\delta$ is directly used as a velocity control input to set the steering wheel velocity $\dot{\delta}$. In practice it could also be possible to use the steering angle δ_k at trajectory state $k = 1$ with position control to control the steering.

Both the online steady-state drifting MPC and trajectory tracking MPC run at 50 Hz, for a horizon of 1 second. We warm-start each successive NLP run with the solution from the previous run, and for the tracking task initialize all variables with the values from the reference trajectories at the corresponding knot points.

The offline parking trajectory optimization typically converges within 1 second for tested trajectory horizons of up to 5 seconds. Notably, as the horizon changes, this directly affects the amount of drifting in the parking maneuver, as also observed in [14].

IV. Results

In this section we present results from using our framework for steady-state drifting as well as for drift parking. The reader is encouraged to watch the accompanying video[1] for visualizations of all presented skidding maneuvers. We stress that our results are only possible due to the presence of the fused kinematic-dynamic model in our optimization framework, and that inserting either standalone model results in no (predicted) drifting for the kinematic model, or non-convergence for the dynamic bicycle model due to the tire slip angle becoming singular at low speeds (i.e., when starting from rest).

A. Steady-State Drifting

Figures 1 (left) and 3 show snapshots from executing steady-state drifting at desired yaw rates ranging from 2 to 5 rad/s.

[1] https://youtu.be/MF1_fS6CQQs

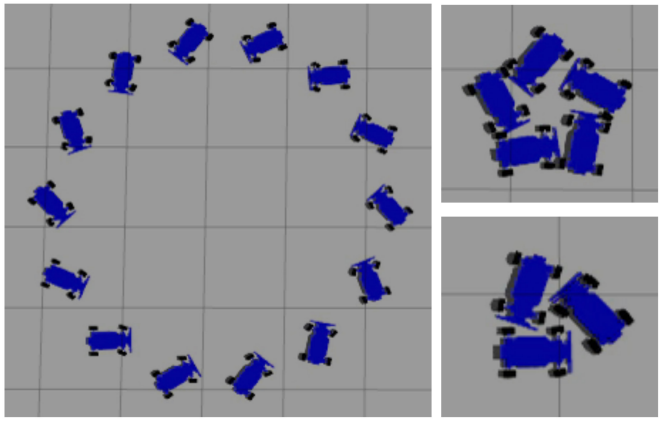

Fig. 3. Clockwise from left: steady state drifting in Gazebo at yaw rates of 2, 4, and 5 rad/s.

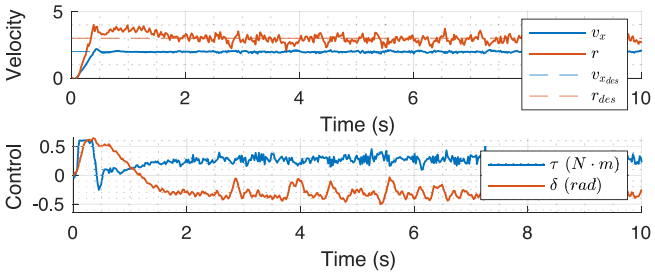

Fig. 4. Autonomous drifting with yaw rate $r = 3$ rad/s and body velocity $v_x = 2$ m/s. For clarity due to scaling, the control signals shown are $\tau_w = F_x w_r/4$ for a single wheel, and the desired steering angle δ.

Figure 4 shows tracking performance for autonomous drifting from rest at a desired yaw rate r_g of 3 rad/s and corresponding desired body velocity v_{x_g} of 2 m/s. The accompanying video illustrates several additional steady-state drifts, ranging from yaw rates of 1 rad/s to 5 rad/s. For these drifts, the cost function is as described in Section Section III-B.1, to minimize the error between the actual and desired body velocity and yaw rate. The vehicle starts at rest in each scenario, where the kinematic model can accurately model the dynamics, before quickly transitioning to the dynamic vehicle part of the fused model. While the tracking of the desired body velocity is very accurate, there are some oscillations present around the desired yaw rates, with possible explanations relating to noisy and varying state estimation, or further needed tuning of cost function weights between desired velocity and yaw rate. We also show additional steady-state drifting examples in the video, and we note that selecting a desired v_{x_g} is optional, as the framework will autonomously converge to drifting at a particular v_x for a desired yaw rate.

To verify the robustness of our framework, each time before querying the MPC, we add uniform random noise in the range of $[-0.35, 0.35]$ to each element of the current car state measurement $\boldsymbol{x}_{\text{init}}$ used to initialize the MPC (Equation (13)). Figure 5 shows the corresponding true simulation states (not used by the MPC), and we observe that although the added state noise results in more variable control signals, the car is still able to steadily drift at the desired rates. Beyond this noise range we observe occasional lapses in continuous drifting.

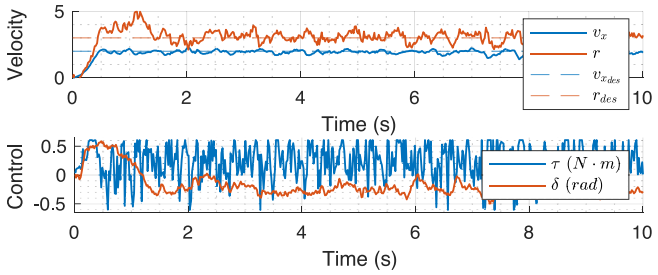

Fig. 5. Autonomous drifting with yaw rate $r = 3$ rad/s and body velocity $v_x = 2$ m/s with uniform random sensor noise in the range of $[-0.35, 0.35]$. For clarity due to scaling, the control signals shown are $\tau_w = F_x w_r/4$ for a single wheel, and the desired steering angle δ.

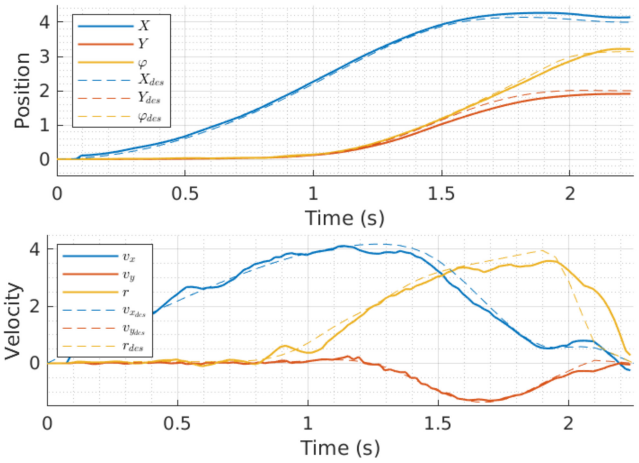

Fig. 6. Drift parking positions X, Y, φ (top) and velocities v_x, v_y, r (bottom) output from the offline trajectory optimization for a goal location of $(X_g, Y_g, \varphi_g) = (4, 2, \pi)$. The desired states are dotted, and the performance of our tracking MPC is shown by the solid lines. The performance is accurate even across the transient drift portion of the maneuver, as shown by the large values in v_y and r.

B. Drift Parking

In this section we consider a dynamic parking maneuver where the car starts from rest at the origin, and has a desired parking goal of $(X_g, Y_g, \varphi_g) = (4, 2, \pi)$. The cost function is as described in Section III-B.2, and the horizon is 2.25 seconds. For such a short horizon, a dynamic parking maneuver is necessary in order to achieve the goal. Due to starting and ending at rest (in valid kinematic model ranges), the fused kinematic-dynamic model is crucial to the success of both the convergence of the optimization as well as of the actual executed motion. The (X, Y, φ) trajectory returned by our optimization framework is shown in blue in Figure 7, and additional states are shown by the dotted lines in Figure 6.

Figure 7 also shows a sample open-loop execution of the trajectory found with the offline optimization framework in green. We note that such open-loop executions deviate from the desired trajectory during the transient drift portion of the maneuver, causing errors in both final position and orientation. The car is likely to either oversteer or understeer, making the final position and orientation unpredictable.

By contrast, by running our tracking MPC as described in Section III-C, we are able to significantly improve on the

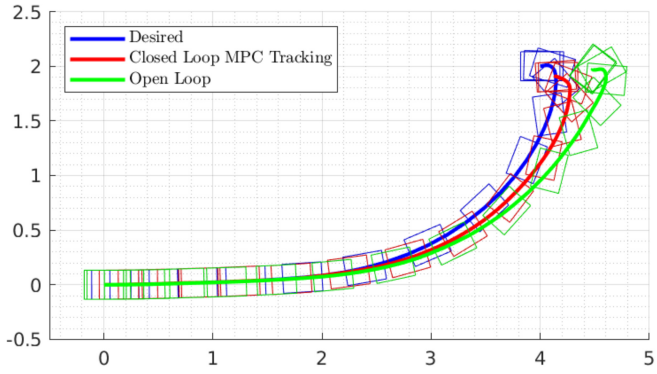

Fig. 7. Drift parking desired trajectory, closed loop tracking with MPC, and an example open loop execution of the offline generated optimal trajectory. Our closed-loop tracking with MPC significantly improves upon the open-loop performance, which is possible due to the fused kinematic-dynamic bicycle model valid at both low and high speeds.

open-loop performance, as shown in red in Figure 7. Notably, as shown in Figure 6, we are able to accurately track the velocity in the vehicle lateral direction v_y, as well as the yaw rate r, even during drifting, which is a large source of error for open-loop execution. In particular, with our framework, we are able to achieve less than 2% error in both position and orientation with the nonlinear MPC, compared with the open-loop performance of 10% error in position and 30% error in orientation.

To verify the robustness of our framework, we additionally run 30 experiments where we randomly initialize the car's starting position $(X, Y) \in [-0.5, 0.5] \in \mathbb{R}^2$ m of the origin while still tracking the same drift parking maneuver. For the final state of the car after tracking the trajectory, we observe a mean position error of 0.030 m with standard deviation of 0.014 m, and a mean orientation error of -0.044 rad with a standard deviation of 0.145 rad. This shows our framework is able to accurately track the transient drifting trajectory under significantly noisy initial conditions, which would not be possible with open-loop executions.

V. Conclusion

In this letter, we proposed using a fused kinematic-dynamic bicycle model in an optimization framework to plan and control dynamic vehicle maneuvers involving drifting. The fused model avoids issues with the singularities of popular tire model approximations such as the Pacejka model, and can thus be used for stop-and-go motions and purely lateral drifting. We note that the motions we planned and controlled in this letter would not be possible if using either standalone kinematic or dynamic bicycle models in the optimization framework. Our nonlinear MPC framework can be run online for steady-state drifting, or offline to generate specific parking maneuvers. The offline generated parking maneuvers can be tracked online with the same framework, which significantly improves on open-loop performance. We also verified the robustness of our framework under noisy state estimation measurements and variable initial position conditions.

Future work will further validate the framework on scaled vehicle hardware. As the success of the planning and control of the model is currently dependent on accurate identification of the Pacejka tire parameters, we also plan to incorporate machine learning to adapt to variable environmental conditions, for example as shown in [11].

References

[1] E. Velenis and P. Tsiotras, "Minimum time vs maximum exit velocity path optimization during cornering," in *Proc. IEEE Int. Symp. Ind. Electron. (ISIE)*, vol. 1. Dubrovnik, Croatia, Jun. 2005, pp. 355–360.
[2] E. Velenis, D. Katzourakis, E. Frazzoli, P. Tsiotras, and R. Happee, "Steady-state drifting stabilization of RWD vehicles," *Control Eng. Pract.*, vol. 19, no. 11, pp. 1363–1376, 2011.
[3] C. Voser, R. Y. Hindiyeh, and J. C. Gerdes, "Analysis and control of high sideslip manoeuvres," *Veh. Syst. Dyn.*, vol. 48, no. sup1, pp. 317–336, 2010.
[4] J. Y. Goh and J. C. Gerdes, "Simultaneous stabilization and tracking of basic automobile drifting trajectories," in *Proc. IEEE Intell. Veh. Symp. (IV)*, Gothenburg, Sweden, Jun. 2016, pp. 597–602.
[5] J. Z. Kolter, C. Plagemann, D. T. Jackson, A. Y. Ng, and S. Thrun, "A probabilistic approach to mixed open-loop and closed-loop control, with application to extreme autonomous driving," in *Proc. IEEE Int. Conf. Robot. Autom.*, Anchorage, AK, USA, May 2010, pp. 839–845.
[6] E. Jelavic, J. Gonzales, and F. Borrelli, "Autonomous drift parking using a switched control strategy with onboard sensors," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 3714–3719, 2017.
[7] F. Zhang, J. Gonzales, K. Li, and F. Borrelli, "Autonomous drift cornering with mixed open-loop and closed-loop control," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 1916–1922, 2017.
[8] F. Zhang, J. Gonzales, S. E. Li, F. Borrelli, and K. Li, "Drift control for cornering maneuver of autonomous vehicles," *Mechatronics*, vol. 54, pp. 167–174, Oct. 2018.
[9] J. H. Jeon, S. Karaman, and E. Frazzoli, "Anytime computation of time-optimal off-road vehicle maneuvers using the RRT*," in *Proc. 50th IEEE Conf. Decis. Control Eur. Control Conf.*, Dec. 2011, pp. 3276–3282.
[10] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Stockholm, Sweden, May 2016, pp. 1433–1440.
[11] M. Cutler and J. P. How, "Autonomous drifting using simulation-aided reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Stockholm, Sweden, 2016, pp. 5442–5448.
[12] T. K. Lau, "Learning autonomous drift parking from one demonstration," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, Karon Beach, Thailand, 2011, pp. 1456–1461.
[13] H. Pacejka, *Tire and Vehicle Dynamics*. Burlington, VT, USA: Elsevier, 2005.
[14] G. Bellegarda and K. Byl, "Versatile trajectory optimization using a LCP wheel model for dynamic vehicle maneuvers," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Paris, France, 2020, pp. 7905–7911.
[15] G. Bellegarda and K. Byl, "Trajectory optimization for a wheel-legged system for dynamic maneuvers that allow for wheel slip," in *Proc. IEEE 58th Conf. Decis. Control (CDC)*, Nice, France, 2019, pp. 7776–7781.
[16] G. Bellegarda and K. Byl, "An online training method for augmenting MPC with deep reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Las Vegas, NV, USA, 2020, pp. 5453–5459.
[17] J. Kabzan *et al.*, "AMZ driverless: The full autonomous racing system," *J. Field Robot.*, vol. 37, no. 7, pp. 1267–1294, 2020.
[18] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," in *Proc. IEEE Intell. Veh. Symp. (IV)*, Seoul, South Korea, 2015, pp. 1094–1099.
[19] J. M. Gonzales, "Planning and control of drift maneuvers with the Berkeley autonomous race car," Ph.D. dissertation, Dept. Mech. Eng., Univ. California, Berkeley, CA, USA, 2018.
[20] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: A software framework for nonlinear optimization and optimal control," *Math. Program. Comput.*, vol. 11, no. 1, pp. 1–36, 2019.
[21] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Math. Program.*, vol. 106, no. 1, pp. 25–57, 2006.
[22] S. Karaman *et al.*, "Project-based, collaborative, algorithmic robotics for high school students: Programming self-driving race cars at MIT," in *Proc. IEEE Integr. STEM Educ. Conf. (ISEC)*, Princeton, NJ, USA, 2017, pp. 195–203.