2015 American Control Conference
Palmer House Hilton
July 1-3, 2015. Chicago, IL, USA

Predictive Trajectory Guidance for (Semi-)Autonomous Vehicles in Public Traffic*

Thomas Weiskircher[1] and Beshah Ayalew[1]

Abstract— **The safe and reliable operation of autonomous and semi-autonomous vehicles in public traffic requires the tight integration of environmental sensing and vehicle dynamics control. In this paper, a predictive control framework is outlined that connects both areas. Specifically, a trajectory guidance module is posed as a nonlinear model predictive controller that computes the optimal future vehicle trajectory using information from environmental sensing for other objects as well as by imposing public traffic rules. It is also sought to minimize the number of vehicle specific parameters needed for the guidance by adopting a particular particle motion description for the vehicle. The computed control input set for the trajectory guidance is passed as a reference for lower-level vehicle dynamics control systems. The definitions of the objective functions and constraints and the adopted vehicle motion model allow for a unified predictive trajectory guidance scheme for fully autonomous and semi-autonomous vehicles in public traffic with multiple dynamic objects. The performance of the proposed scheme is illustrated via simulations of an autonomous and a semi-autonomous vehicle in a few traffic scenarios such as intersections and collision avoidance. Execution time considerations are also analyzed.**

I. INTRODUCTION

Autonomous and semi-autonomous controlled vehicles ((S)ACVs) are under active research and development to realize their potential for providing safe, efficient and sustainable mobility solutions. One aspect of the control framework for (S)ACVs covers trajectory planning to guarantee a collision-free optimal path under the constraints of prevailing external conditions such as road boundaries and traffic regulations. The existing research in this area generally falls in one of two categories: non-reactive path planning and reactive path planning. The works in first category often rely on simple motion principles, geometric optimization, graph search methods or heuristics for special maneuvers e.g. finding a route in uncertain terrain. These approaches may result is suboptimal solutions as the assumptions made at the time of trajectory computation may change during the maneuver, for example, when an obstacle object changes its direction suddenly. By contrast, the works in the second category often use Model Predictive Control (MPC) or resampling methods, which could accommodate such dynamic changes. However, they involve a large computational burden as they typically solve a linear or nonlinear program at each control update, and they need high update rates to be compatible with the system dynamics. These challenges are being addressed with recent developments in the field of real-time optimization algorithms. It has been shown that it is possible to solve even nonlinear programs in a suitable time scale for systems with dynamics in millisecond range using multi-shooting methods and optimized auto-generated code [1]–[3].

There has been an extensive volume of work in the application of MPC to vehicle control, specially in the area of vehicle dynamics control and collision avoidance. Some of these works focus on collision avoidance for (semi-)autonomous systems via single control inputs such as active front steering without longitudinal vehicle control [4], [5]. Others re-framed the dynamic model for MPC in terms of the arc length of the path to overcome the time dependencies of references and constraints [6], [7]. MPC has also been applied for low-level trajectory tracking and the development of controllers for collision avoidance considering robustness to uncertainties in adopted driver models [8], [9]. Some of the mentioned works combine an ideal/simplified trajectory planning module with trajectory tracking modules that use high fidelity nonlinear vehicle dynamics models. These models require a large set of vehicle parameters (e.g. tire data, vehicle mass and inertia) and high update rates. Furthermore, to reduce the computational requirements, linearized models are often used for the MPC task assuming that the operating point of the vehicle is not changing in the prediction horizon, see e.g. [10], [11]. While the execution time may be reduced with linearization, these assumptions do not necessarily hold in events and vehicles requiring combined lateral and longitudinal control over long prediction horizons; the results are at best suboptimal.

Still, in general, it is not recommended to use high fidelity vehicle dynamics models for MPC. This is because the computational burden of the resulting nonlinear programming problem is estimated to be $O\left(n_x^2+n_y^2+n_p\right)$ where n_x is the model order or number of states, n_y is the number of outputs and n_p the number of parameters [12]. Thus, low complexity vehicle models are a focus of recent research for MPC-based trajectory planning [13], [14]. Each of these works use slightly different simplified particle or point mass models for the target vehicle. These kinds of models have also been used in non-predictive trajectory planning algorithms [7], [15].

In this paper, we build on the above works and design a control framework for (semi-)autonomous vehicles that poses the trajectory planning *and* control function, or for short, the trajectory guidance, as a nonlinear model predictive controller. This trajectory guidance module works with environmental sensing, traffic and navigation information that can be

*This work was supported by a fellowship within the Postdoc-Program of the German Academic Exchange Service (DAAD)

[1]Thomas Weiskircher and Beshah Ayalew are with the Applied Dynamics & Control Research Group at the Clemson University - International Center for Automotive Research, 4 Research Drive, 29607, Greenville, SC, USA, {tweiski, beshah}@clemson.edu

utilized for (S)ACV operation in public street scenarios with multiple dynamic objects.The framework retains traditional vehicle dynamics control systems for reference tracking using steering, braking or powertrain actuators. The predictive controller uses a formulation of the particle motion model for the vehicle in curvilinear path coordinates, with both longitudinal and lateral control inputs. The adopted modeling and problem formulation avoids the need for detailed vehicle data/parameters, helps with the computational aspects, and unifies the resulting vehicle control framework so that it can be used interchangeably for semi-autonomous or fully autonomous vehicles.

The rest of the paper is organized as follows. In Section II, we introduce the proposed control structure, present the reduced vehicle motion model and detail the predictive control framework. This includes the definitions of the control objectives and the constraints for the trajectory guidance as well as a brief discussion of the low-level vehicle dynamics control systems. Section III illustrates some operating modes of the algorithm using simulations of fully autonomous and semi-autonomous driving. It also includes some analysis of algorithm execution times. The paper ends by summarizing the conclusions of the work in Section IV.

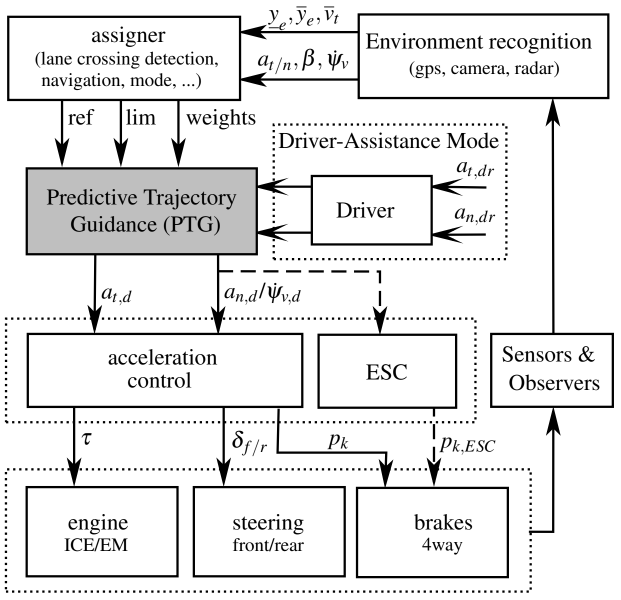

Fig. 1. Multi-level control structure for (semi-)autonomous driving

II. Predictive Control Framework and Reduced Vehicle Motion Model

Figure 1 depicts the proposed control structure for the (semi-)Autonomous Controlled Vehicle (S)ACV. The assigner module processes information from the environmental recognition and vehicle dynamics sensors and delivers all required data about road lane boundaries and objects to be avoided by the (S)ACV. The (S)ACV's position and alignment to the road is assumed to be measured or estimated. The predictive trajectory guidance (PTG) module is a nonlinear MPC that calculates the longitudinal and lateral accelerations that are required to follow an optimal reference path within the speed limit specified in traffic regulations. For semi-autonomous operation, additional references generated by the human driver may be taken into account in the trajectory guidance module. The control output of the PTG is directly transferred to lower-level vehicle dynamics control systems such as engine/electric motor torque controls τ, individual brake pressures p_j, or front/rear steering angles δ_f, δ_r. As an option, an electronic stability control (ESC) module may also be considered but it is omitted for the discussions in this paper.

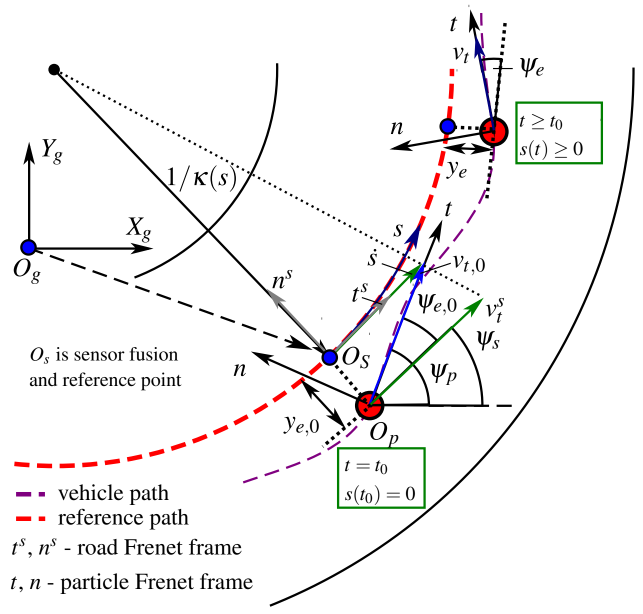

Fig. 2. Definitions of the curvilinear motion of the vehicle

A. Particle Motion Description

As mentioned in the introduction, for computational feasibility of the PTG on real-time hardware, we adopt a particle kinematics model to describe the vehicle's gross motion. A 2D curvilinear motion of a particle is described in the Frenet frame with the definitions depicted in Fig. 2 and also discussed in [7], [13]. The motion of the particle with respect to the local reference path/lane centerline is given by the angular alignment error ψ_e and lateral error y_e. It can be shown that the motion can be described by the following equations:

$$\dot{v}_t = a_t, \tag{1a}$$

$$\dot{\psi}_e = \dot{\psi}_p - v_t \cos(\psi_e) \left(\frac{\kappa(s)}{1 - y_e \kappa(s)} \right), \tag{1b}$$

$$\dot{y}_e = v_t \sin(\psi_e), \tag{1c}$$

$$\dot{a}_t = 1/T_{a_t} \left(a_{t,d} - a_t \right), \tag{1d}$$

$$\ddot{\psi}_p = 1/T_{\dot{\psi}_p} \left(\dot{\psi}_{p,d} - \dot{\psi}_p \right), \tag{1e}$$

$$\dot{s} = v_t \cos(\psi_e) \left(\frac{1}{1 - y_e \kappa(s)} \right). \tag{1f}$$

In this description, the desired acceleration $a_{t,d}$ and the desired yaw rate $\dot{\psi}_{p,d}$ are used as inputs to maneuver the particle/vehicle along a given reference path. The reference path curvature $\kappa(s)$ is assumed to be known along the reference path coordinate s and v_t, a_t are the particle speed and acceleration along its path. $\dot{\psi}_p$ is the yaw rate, ψ_e is the aligning error to the reference path, and $T_{a_t}, T_{\dot{\psi}_p}$ are the time constants of the first-order approximation of the longitudinal and lateral actuation dynamics for the (S)ACV. $\dot{s}$ is the speed projected on the reference path as shown in Fig. 2. It is obvious that the particle motion model above includes only two vehicle specific parameters ($T_{a_t}, T_{\dot{\psi}_p}$).

In (1e) the desired yaw rate is used to control the direction of the particle. With the knowledge of the curvature of the reference path $\kappa(s)$ and the assumption that the (S)ACV is required to follow this path, this control input can be substituted for by:

$$\dot{\psi}_{p,d} = \dot{\psi}_{p,r} + \Delta\dot{\psi}_{p,d} = v_t \kappa(s) + \Delta\dot{\psi}_{p,d}. \tag{2}$$

Since the goal is to control the particle/vehicle motion along the reference path, the predictive trajectory guidance (PTG) module only needs to control the deviation with the new input $\Delta\dot{\psi}_{p,d}$.

B. Objects, Constraints and MPC Formulation

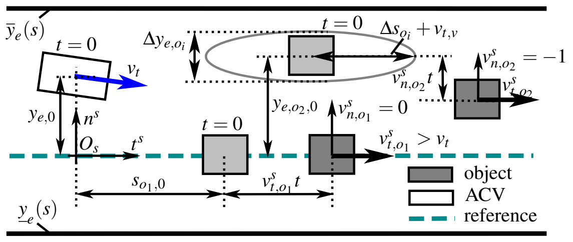

Fig. 3. Object motion definition ($\kappa = 0$) in road reference frame

To incorporate a description of other objects (obstacles or other vehicles) on the road, the particle motion description of other objects in the road frame $O_s(t^s, n^s)$ are given by:

$$s_{o_i} = s_{o_i,0} + v^s_{t,o_i} x_t + \frac{1}{2} a^s_{t,o_i} x_t^2, \tag{3a}$$

$$y_{e,o_i} = y_{e,o_i,0} + v^s_{n,o_i} x_t + \frac{1}{2} a^s_{n,o_i} x_t^2, \tag{3b}$$

$$\dot{x}_t = 1, \tag{3c}$$

with the additional state x_t in the MPC model, that represents the internal time. x_t helps to calculate the object position by means of the parameters $v^s_{t,o_i}, v^s_{n,o_i}, a^s_{t,o_i}, a^s_{n,o_i}$ only. We assume that all objects follow the road reference path of the ACV with constant longitudinal and lateral accelerations starting at their position with their initial speed at the beginning of the prediction, see Fig. 3. For each object i, this simple kinematic model describes its motion in the prediction horizon starting at the initial point $(s_{o_i,0}, y_{e,o_i,0})$. Now, the (S)ACV needs to stay away from each object with a safe distance/margin. This requirement is defined with the following elliptical hard constraints:

$$1 \leq \left(\frac{\Delta y_{e_i}}{\Delta y_{e,o_i}}\right)^2 + \left(\frac{\Delta s_{e_i}}{\Delta s_{o_i}}\right)^2 = d_{o_i}, \tag{4a}$$

$$\Delta s_{e_i} = s - s_{o_i}, \tag{4b}$$

$$\Delta y_{e_i} = y_e - y_{e,o_i}, \tag{4c}$$

$$\Delta s_{o_i} = \Delta s_{o,ss} + \zeta_{Do}, \tag{4d}$$

$$0 \leq \zeta_{Do}. \tag{4e}$$

as depicted in Fig. 3. Thus, the (S)ACV has the speed dependent minimum distance along the reference path (e.g. the road lane) of $\Delta s_{o_i} + \zeta_{Do}$ and a lateral distance adjusted by $\Delta y_{e,o_i}$. For traffic objects such as cars, $\Delta y_{e,o_i}$ is calculated by incorporating the geometry of the object and the (S)ACV. ζ_{Do} is a slack variable which allows the solver to find a feasible solution in emergency situations. With these constraints, the (S)ACV is required to be in a different lane from an object vehicle *or* to keep a minimum distance from an object in the same lane.

The constraints for the lane boundaries and the speed limit (e.g. from traffic regulations) are assumed to be modeled or given as known (polynomial) functions of the reference road coordinate s. Then, the hard constraints for the speed limit and the lateral road limits read as follows:

$$\Delta\overline{v}_t = \overline{v}_t(s) - v_t \geq 0, \tag{5a}$$

$$\Delta\overline{y}_e = \overline{y}_e(s) - y_e \geq 0, \tag{5b}$$

$$\Delta\underline{y}_e = y_e - \underline{y}_e(s) \geq 0, \tag{5c}$$

$$\left(v_t\left(\dot{\psi}_{p,d} v_t + \Delta\dot{\psi}_{p,r}\right)/\overline{a}_t\right)^2 + \left(a_{t,d}\right)^2 \leq \left(\mu_H g - \zeta_{gg}\right)^2. \tag{5d}$$

Eq. (5d) defines the acceleration limits with respect to the absolute value $\mu_H g$ of the friction ellipse of a real vehicle, the scaling of the ellipse via $\overline{a}_t \leq 1$, the tire-road friction coefficient μ_H, and the gravitational constant g. A slack variable ζ_{gg} enables the formulation of the limit value of the combined accelerations as a soft constraint. Furthermore, the longitudinal acceleration is limited to its upper ($\overline{a}_{t,d}$) and lower bounds ($\underline{a}_{t,d}$):

$$\underline{a}_{t,d} \leq a_{t,d} \leq \overline{a}_{t,d}, \tag{6a}$$

$$\underline{a}_{n,d} \leq a_{n,d} \leq \overline{a}_{n,d}. \tag{6b}$$

At this point, two more hard constraints are added. The first one prevents the singularity in the model definitions (1f) and (1b) for $y_e \kappa(s) = 1$, which is not normally experienced with common combinations of reference path curvatures and lateral errors occurring in public roads. The second constraint restricts the algorithm not to turn the vehicle at low speed (below the minimum turning radius of the vehicle):

$$(v_t \kappa(s) + \Delta\dot{\psi}_p) \leq v_t \overline{\kappa}, \tag{7a}$$

$$\overline{\kappa} = \frac{1}{(t_v/2 + l/sin(\overline{\delta}_f))}. \tag{7b}$$

Herein, the vehicle's track width t_v, wheel base l and limit of the front steering angle $\overline{\delta}_f$ are required.

Finally, a constraint on the arc length s allows the guidance module to stop the vehicle at a certain point along the reference path, while constraints on the slack variable allow easy tuning of the soft constraint (5d):

$$s \leq \bar{s}, \tag{8a}$$

$$0 \leq \zeta_{gg} \leq \overline{\zeta}_{gg}. \tag{8b}$$

The performance index of the MPC algorithm of the PTG module weighs the tracking and control errors as follows:

$$J = \sum_{k=0}^{N_p} \underbrace{||y_k - r_k||_Q^2}_{\text{tracking error}} + \sum_{k=0}^{N_p-1} \underbrace{||u_k - u_r||_R^2}_{\text{control minimization}} \tag{9}$$

Here, k is the prediction step $k \in (0,1,2,\cdots,N_p)$ and N_p is the prediction length. The latter results from the sample and update time ΔT of the MPC and is related to the prediction horizon H_p: $N_p = H_p/\Delta T$. The nonlinear program to be solved at each MPC update is then given by:

$$\min_{u_k} J(y_k, u_k) \tag{10a}$$

$$where: \; x_k = \begin{bmatrix} v_t & y_e & \psi_e & s & a_t & \dot{\psi}_p & x_t \end{bmatrix}^T \tag{10b}$$

$$u_k = \begin{bmatrix} a_{t,d} & \Delta\dot{\psi}_{p,d} \end{bmatrix} \tag{10c}$$

$$y_k = \begin{bmatrix} y_e & v_t & \zeta_{gg} & \zeta_{Do} - v_t \end{bmatrix}^T \tag{10d}$$

$$r_k = \begin{bmatrix} y_{e,r} & v_{t,r}^v & \zeta_{gg,r} & e_{\zeta_{do},r} \end{bmatrix}^T \tag{10e}$$

$$s.\,t.: \dot{x}_t = f(x_k, u_k), \tag{10f}$$

where x_k, y_k are the equidistantly sampled states and outputs of the continuous system (10f), obtained by applying the piecewise constant control inputs u_k calculated by the MPC algorithm. The object distance slack error reference is selected with $e_{\zeta_{do},r} = 0$.

Remark 1: It should be noted that $\Delta\dot{\psi}_p$ of u_k is transformed back to the original control input by rearranging (2) before it is used for further operations in the lower-level control.

Remark 2: The reference values in r_k are taken as constants for the prediction horizon. In case that the speed is required to be less than the reference value $v_{t,r}$ at some point in the prediction horizon, it is restricted with the path dependent limit (5a). The references for u_r are set to zero.

Remark 3: By directly including the reference path coordinate s as a dynamic state of the model, we avoid the need for a spatial re-formulation of the references, the dynamic model and the constraints as done in some literature [6], [7]. This enables us to retain time as the independent variable for the MPC prediction model.

C. Lower-Level Control Systems

The PTG described above uses the longitudinal acceleration and yaw rate in (10c) to optimally control the particle motion according to the objective and constraints defined above. In contrast to classical trajectory planners, which provide the planned state trajectory (speed and position) as references for the lower-level control systems, here the control inputs of the PTG (the longitudinal acceleration and yaw rate) are passed as references for the lower-level control systems. The PTG itself is set up as a feedback controller (MPC) that minimizes the actual vehicle tracking error $(y_k - r_k)$ by injecting the first optimal control input $(k = 0)$ until the next update.

The lower-level control modules will not be outlined here in much detail for lack of space. Some possible configurations are presented in [16]. However, many traditional vehicle dynamics control systems could be adopted for this purpose. The version we use here is a speed dependent gain-scheduled controller for tracking the lateral (yaw rate) reference via front steering, and a feed-forward plus a feedback PI controller for tracking the longitudinal acceleration via traction and braking forces.

III. Algorithm Discussions and Results

This section includes discussions of the several degrees of freedom incorporated in the proposed PTG framework that can be set to achieve desired operating modes and performance levels. For the simulations included and discussed in this section, the PTG and the lower-level controllers are applied to a higher-fidelity nonlinear vehicle model (which includes Pacejka tire force model, tire force relaxation, load transfer and actuator dynamics for steering, engine, and brakes). The following MPC parameters are found to give sufficiently good results for the vehicle simulated: $H_p = 4$ s, $N = 40$ and so MPC update interval of $\Delta T = 0.1$ s. Also, oversampling of the MPC at $T_{mpc} = 0.03$ s is found necessary to generate smooth control trajectories.

A. PTG Modes and Parameter Selection

Following the definition of the cost function (9), the weightings Q and R of the output and control variables are useful to prioritize different objectives. Here, we discuss two main operating modes: Fully Autonomous (FA) and Semi-Autonomous (SA). For the FA mode, path following along the middle of a lane $(y_{e,r} = 0)$ is combined with tracking a reference velocity $v_{t,r}$ using longitudinal and lateral control inputs. In the SA mode, the PTG is restricted to one control input only as discussed in the following paragraph. In addition, in FA mode, while the vehicle is desired to follow the reference path whenever possible, a deviation y_e may appear as an optimal trade-off between acceleration levels, speed and path tracking. The slack weights $Q_{\zeta_{gg,Do}}$ are set high to restrict the usage of the full vehicle potential and generate comfortable trajectories most of the time. In safety related situations, e.g. for collision avoidance, the underlying optimization solver reduces the soft limit automatically when hard constraints like distance d_{o_i} to objects force it to do so, in order to generate a feasible solution.

To separate the SA from the FA mode, the hard limits for the control inputs in (6) are chosen differently. We introduce two example SA modes. In the first, Adaptive Cruise Control (ACC) mode, the PTG is only allowed to use the longitudinal acceleration to control the vehicle speed. In this mode, it prevents collisions by reducing the speed and following objects in front in a distance defined by (4). The second SA mode is named Collision Avoidance (CA) and Lane Keeping

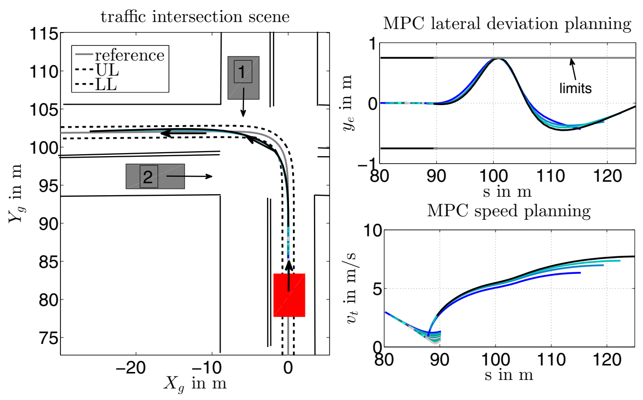

Fig. 4. Intersection: overview and PTG output for speed and lateral deviation

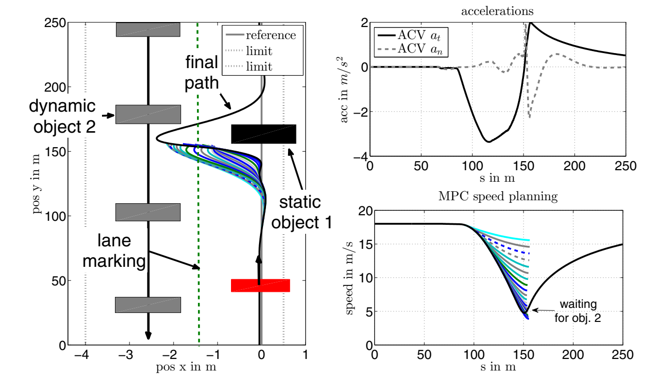

Fig. 6. FA CA and passing: control inputs, accelerations and speed planning

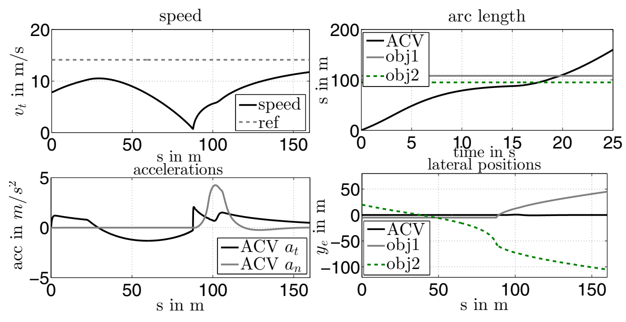

Fig. 5. Intersection: path, speed, and accelerations of ACV and positions

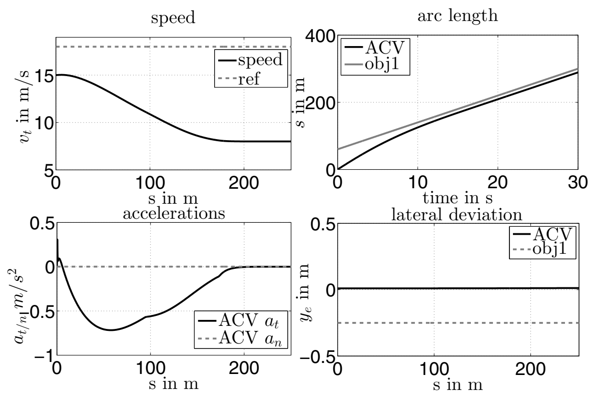

Fig. 7. SA ACC mode: positions, lateral deviation, accelerations and speed

Assist (LKA), and reduces the PTG to control only the lateral motion of the vehicle. Thus, it will pass slower objects and it tries to follow the reference trajectory with the speed being controlled by a human driver.

B. Simulation Results - FA Mode

Two results are given for the FA mode. First, the ACV needs to follow a reference path through an intersection with a left turn in the presence of two objects (other vehicles), see Fig. 4 and 5. A traffic light detected by the environmental sensors leads to a restriction of the vehicle arc length while object no. 2 from the left side is allowed to pass the intersection first. Thus, the PTG starts reducing the speed at about $s = 30$ m. Then, at t = 10 s the traffic light is set to green and the arc length restriction is softened to allow the PTG to accelerate the ACV (at this time, the ACV has reached $s = 90$ m). Before crossing the intersection and tracking the reference path, the PTG makes the ACV slow down to a crawl until object 1 passes the intersection. Herein, the black line is the trajectory of the ACV.

The next results are generated for two objects plus the ACV in a collision avoidance (CA) scenario: one slow object in the same lane initially in front of the ACV and another object traveling in the opposite direction on a second lane. Fig. 6 depict the results. As the reference speed of the ACV is 18 m/s but object 1 in front is standing still, the ACV needs to pass object 1 to follow the reference speed. To do so, the PTG first reduces the speed and waits until object 2 on the left lane has passed. Then, the PTG accelerates the ACV to the reference speed again and forces the ACV to pass object 1 by adding some lateral acceleration to the control inputs. Fig. 6 also shows the detailed calculations of the MPC for the vehicle speed and lateral deviation planning.

C. Simulation Results - SA Modes

The first SA case considered is ACC mode in straight-line driving showing in Fig. 7. In this case, the PTG controls the vehicle longitudinal acceleration a_t to track the vehicle reference speed, and it reduces the accelerations to prevent crashing with the object or vehicle in front of the (S)ACV. Considering the same maneuver conditions but with the PTG in CA & LKA modes, the results depicted by Fig. 8 are obtained. Note that the speed is constant as the human driver is not reacting when the (S)ACV approaches the object/vehicle in front (see arc length). Then, the PTG uses the lateral control input $\Delta\psi_p$ to pass the object vehicle in a safe distance of about 2 m until it tracks the reference with zero-deviation. The PTG makes the (S)ACV pass on the left of the object as the traffic lane constraint prevents it from doing so on the right side.

D. Execution Time of MPC

The execution times (ET) of the MPC for the PTG were recorded in the previously discussed maneuvers. Table I

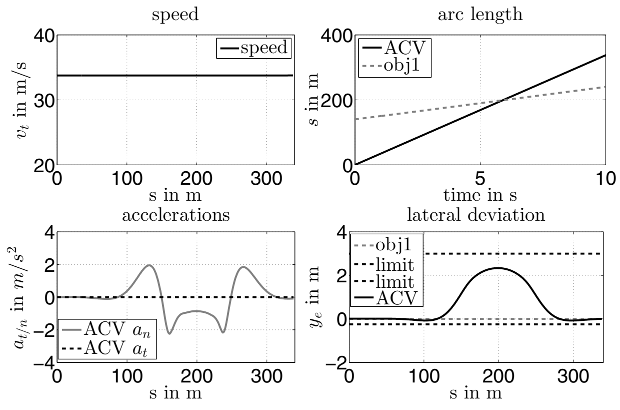

Fig. 8. SA CA & LKA mode: lateral deviation, accelerations and speed

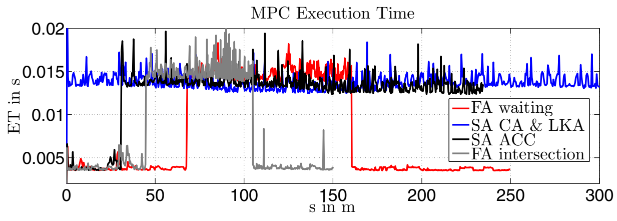

Fig. 9. Execution time for different maneuvers over maneuver arc length

depicts the mean ET, the minimal and the maximum ET estimated by the solver qpOASAS which was used to implement the proposed nonlinear MPC scheme. It can be seen that the mean execution time is fairly constant in the range of 8-13 ms, while the minimum ET for FA mode is less than for the other modes. The ET depicted in Fig. 9 consists of certain piecewise constant levels. A close examination along with the previous results indicates that the ET jumps to a higher level when some inequality constraint is activated, e.g. it is easy to conclude this from the SA ACC maneuver or the intersection maneuver, when approaching objects. Active inequality constraints engaged with the Active-Set method of the solver are the reason for this behavior. This is also obvious from the SA CA & LKA mode which is limited in speed by the acceleration limits. While further investigation is required for a conclusive remark, these results seem to indicate that it is quite possible to implement the proposed framework in modern real-time hardware.

TABLE I

EXECUTION TIME ON INTEL I7 4600U NOTEBOOK CPU, 2.7 GHZ

scene	FA CA	FA intersection	SA ACC	SA CA & LKA
mean	8.60	10.64	12.65	13.73
min	3.46	3.57	3.54	12.44
max	18.29	32.70	19.63	18.83

IV. CONCLUSIONS

In this paper, a versatile control structure is presented for implementing a predictive trajectory guidance module (PTG) for fully autonomous and semi-autonomous vehicle trajectory planning and control. The PTG uses a particle motion model for the vehicle dynamics expressed in curvilinear coordinate frames, as well as several references and hard constraints imposed by an assigner module and vehicle dynamics constraints such tire-road traction capability. The PTG implements a nonlinear MPC scheme to compute the longitudinal and yaw rate targets that give optimal trajectories for (semi-) autonomous controlled vehicles under these constraints. The algorithm shows good performance in various scenarios with dynamic objects representing public traffic scenarios. The PTG scheme is readily configurable for fully autonomous or semi-autonmous operating modes, while requiring very few vehicle-specific parameters. It also offers reduced computational times of the order compatible with modern real-time hardware.

REFERENCES

[1] M. Vukov, A. Domahidi, H. J. Ferreau, M. Morari, and M. Diehl, "Auto-generated Algorithms for Nonlinear Model Predictive Control on Long and on Short Horizons," in *Proceedings of the 52nd Conference on Decision and Control (CDC)*, 2013.

[2] H. J. Ferreau, T. Kraus, M. Vukov, W. Saeys, and M. Diehl, "High-speed moving horizon estimation based on automatic code generation," in *51th IEEE Conference on Decision and Control (CEC)*, 2012.

[3] B. Houska, H. Ferreau, and M. Diehl, "An Auto-Generated Real-Time Iteration Algorithm for Nonlinear MPC in the Microsecond Range," *Automatica*, vol. 47, no. 10, pp. 2279–2285, 2011.

[4] T. Keviczky, P. Falcone, F. Borelli, and J. A. Horvat, "Predictive control approach to autonomous vehicle steering," in *American Control Conference*, Minnesota, USA, 2006, pp. 4670–4675.

[5] P. Falcone, F. Borelli, J. Asgari, H. E. Tseng, and D. Hrovat, "Predictive active steering control for autonomous vehicle systems," *IEEE Control Systems Technology*, vol. 15, no. 3, pp. 566–580, 2007.

[6] Y. Gao, A. Gray, J. Frasch, T. Lin, H. E. Tseng, J. K. Hedrick, and F. Borelli, "Spatial predictive control for agile semiautonomous ground vehicles," *11th AVEC conference*, 2012.

[7] M. Werling and L. Groell, "Low-level controllers realizing high-level decisions in an autonomous vehicle," in *IEEE Intelligent Vehicles Symposium*, 2008.

[8] A. Gray, Y. Gao, J. K. Hedrick, and F. Borelli, "Robust predictive control for semi-autonomous vehicles with an uncertain driver model," in *Proc. of the IEEE Intelligent Vehicles Symposium*, Gold Coast, Australia, 2013, pp. 208–213.

[9] Y. Gao, A. Gray, H. E. Tseng, and F. Borelli, "A tube-based robust nonlinear predictive control approach to semiautonomous ground vehicles," *Vehicle System Dynamics*, vol. 52, no. 6, pp. 802–823, 2014.

[10] V. Turri, A. Carvalho, H. Tseng, K. H. Johansson, and F. Borelli, "Linear model predictive control for lane keeping and obstacle avoidance on low curvature roads," in *Proceedings of the 16th International IEEE Annual Conference on Intelligent Transportation Systems*, The Hague, The Netherlands, 2013, pp. 378–383.

[11] A. Carvalho, Y. Gao, A. Gray, H. E. Tseng, and F. Borelli, "Predictive control of an autonomous ground vehicle using an iterative linearization approach," in *Proceedings of the 16th International IEEE Annual Conference on Intelligent Transportation Systems*, The Hague, The Netherlands, 2013, pp. 2335–2340.

[12] H. J. Ferreau, "Model predictive control algorithms for applications with millisecond timescales," PHD Thesis, KU Leuven, Arenberg Doctoral School of Science, 2011.

[13] G. Prokop, "Modeling human vehicle driving by model predictive online optimization," *International Journal of Vehicle Mechanics and Mobility : Vehicle System Dynamics*, vol. 35, no. 1, pp. 19–53, 2001.

[14] P. Falcone, F. Borrelli, H. E. Tseng, J. Asgari, and D. Hrovat, "A hierarchical model predictive control framework for autonomous ground vehicles," in *American Control Conference*, 2008, pp. 11–13.

[15] M. Werling and J. Ziegler, "Optimal trajectory generation for dynamic street scenarios in a frenet frame," in *International Conference on Robotics and Automation*, Alaska, USA, 2010, pp. 987–993.

[16] T. Weiskircher and B. Ayalew, "Predictive trajectory guidance for (semi-)autonomous vehicles in public traffic: Part 2 - the lower level controllers," in *American Control Conference*, 2015.