# Design and Comparative Analysis of Several Model Predictive Control Strategies for Autonomous Vehicle Approaching a Traffic Light Crossing

**Cvok, Ivan; Pavelko, Lea; Škugor, Branimir; Deur, Joško; Tseng, H. Eric; Ivanovic, Vladimir**

*Source / Izvornik:* **Energies, 2023, 16, 2006 - 2026**

**Journal article, Published version**
**Rad u časopisu, Objavljena verzija rada (izdavačev PDF)**

https://doi.org/10.3390/en16042006

*Permanent link / Trajna poveznica:* https://urn.nsk.hr/urn:nbn:hr:235:049423



*energies*

MDPI

*Article*

# Design and Comparative Analysis of Several Model Predictive Control Strategies for Autonomous Vehicle Approaching a Traffic Light Crossing

**Ivan Cvok [1], Lea Pavelko [1], Branimir Škugor [1,*], Joško Deur [1], H. Eric Tseng [2] and Vladimir Ivanovic [2]**

[1] University of Zagreb, Faculty of Mechanical Engineering and Naval Architecture, 10000 Zagreb, Croatia
[2] Ford Motor Company, Dearborn, MI 48124, USA
* Correspondence: branimir.skugor@fsb.hr

**Abstract:** Recent advancements in automated driving technology and vehicle connectivity are associated with the development of advanced predictive control systems for improved performance, energy efficiency, safety, and comfort. This paper designs and compares different linear and nonlinear model predictive control strategies for a typical scenario of urban driving, in which the vehicle is approaching a traffic light crossing. In the linear model predictive control (MPC) case, the vehicle acceleration is optimized at every time instant on a prediction horizon to minimize the root-mean-square error of velocity tracking and RMS acceleration as a comfort metric, thus resulting in a quadratic program (QP). To tackle the vehicle-distance-related traffic light constraint, a linear time-varying MPC approach is used. The nonlinear MPC formulation is based on the first-order lag description of the vehicle velocity profile on the prediction horizon, where only two parameters are optimized: the time constant and the target velocity. To improve the computational efficiency of the nonlinear MPC formulation, multiple linear MPCs, i.e., a parallel MPC, are designed for different fixed-lag time constants, which can efficiently be solved by fast QP solvers. The performance of the three MPC approaches is compared in terms of vehicle velocity tracking error, root-mean-square acceleration, traveled distance, and computational time. The proposed control systems can readily be implemented in future automated driving systems, as well as within advanced driver assist systems such as adaptive cruise control or automated emergency braking systems.

**Keywords:** automated driving; autonomous vehicle; traffic light crossing; model predictive control; nonlinear control; assessment



**Citation:** Cvok, I.; Pavelko, L.; Škugor, B.; Deur, J.; Tseng, H.E.; Ivanovic, V. Design and Comparative Analysis of Several Model Predictive Control Strategies for Autonomous Vehicle Approaching a Traffic Light Crossing. *Energies* **2023**, *16*, 2006. https://doi.org/10.3390/en16042006

Academic Editor: Balázs Németh

Received: 4 January 2023
Revised: 10 February 2023
Accepted: 13 February 2023
Published: 17 February 2023



## 1. Introduction

In 2019, traffic congestion caused 8.8 billion hours of extra travel time and 3.3 billion gallons of wasted fuel for drivers in the USA, where the congestion and idling time at signalized intersections made up for a large portion of those numbers [1]. Approaching a signalized intersection is a complex traffic scenario, which often turns into a congested bottleneck due to the lack of coordination between vehicles and infrastructure, and sudden acceleration or deceleration maneuvers or idling at a traffic light increase energy consumption and the risk of rear-end collisions [2]. In order to improve vehicle and overall transport system performance in terms of driving performance, safety, comfort, and energy efficiency, several predictive control strategies relying on infrastructure-to-vehicle (I2V) communications have been developed recently [3,4].

A vehicle trajectory planning system for approaching a traffic light is often referred to as a green-light optimal speed advisory (GLOSA) system, which utilizes traffic light signal (TLS) information to find an optimal vehicle speed profile to avoid stopping and idling at traffic lights [5]. It is shown in [6] that a model predictive controller (MPC) having a control horizon that sees two traffic lights ahead reduces energy consumption by 26% at the expense of a trip time increase of only 1% when compared to a controller with no TLS information.

The authors in [7] exploit the knowledge of an upcoming traffic light location and real-time traffic flow data to anticipate possible upcoming vehicle speed profiles, and thus robustly adapt an energy management strategy of a hybrid electric vehicle. When compared to the control strategy with fixed (nonadapted) parameters, fuel consumption savings between 8 and 11% are reported therein. On the other hand, alternative approaches assuming a lack of TLS information and infrastructure-to-vehicle (I2V) communication rely only on statistical learning from driving data to achieve smooth and energy-efficient driving in an average sense. For instance, the authors in [8] use Gaussian processes as a machine learning method to facilitate eco-driving while approaching signalized intersections with traffic lights, which, in addition to the vehicle traveling time, also aims to minimize fuel consumption.

In general, when considering a vehicle approaching a signalized crossing, the objective is to provide an energy-efficient and timely arrival at the green light with minimal use of braking, maintain a safe distance between vehicles, and cruise at or near a set speed [9]. This can be achieved by using different approaches, such as dynamic programming (DP) [10,11], model predictive control [12,13], sequential convex optimization [14], or reinforcement learning [15]. These approaches often include fuel and/or electricity consumption models to obtain energy-efficient vehicle speed trajectories, which increases the dimensionality of the optimization problem and computational complexity. Additionally, due to the switching nature of traffic light states and related constraints, the optimization problem turns into a mixed-integer problem, which demands significant computational effort, and the solution is not guaranteed to be globally optimal. This issue can be addressed in different ways, e.g., by assuming that the vehicle is going to pass the crossing before the next red traffic light state [16]. However, this is not always the case, especially when considering multiple sequential traffic lights [17]. These challenges call for special attention in formulating the optimization problem to ensure desired performance and constraint satisfaction.

A number of control strategies for the GLOSA problem involve separating the problem into multiple steps to obtain the optimal vehicle speed trajectory. A two-step approach is proposed in [18] for the mixed-integer problem, where the problem is first solved without the traffic-light-related integer constraint, and then solved again in the second step if the optimal vehicle intersection arrival time from the first step occurs during the red-light time interval. In the second step, the problem is solved twice, where the vehicle passing time is fixed to either before or just after the red-light time interval detected previously, and the objective is to minimize energy consumption. The authors in [17] also separate the problem into two steps, where a high-level algorithm first determines the values of desired arrival time for each traffic light, and then the optimal velocity profile is calculated in the second step. A similar hierarchical strategy is proposed in [19], where the higher layer of the strategy determines feasible and suitable arrival time values and velocity profiles for each traffic light based on the desired vehicle speed and TLS information, while the lower layer combines the Pontryagin minimum principle and a model predictive control framework to obtain an optimal vehicle velocity trajectory. In [20] a velocity pruning algorithm is performed to obtain the minimum and maximum feasible crossing times for each existing traffic light while accounting for speed limits and TLS information. Then, a weighted directed acyclic graph is constructed, where the nodes represent crossing times in the feasible green-light phases, while the edges are weighted by energy cost to travel along the path. Dijkstra's algorithm is used to obtain an energy-optimal path from the origin, which is the current vehicle position to the destination, i.e., the position of the traffic light. An approach based on the best interpolation in a strip method is proposed in [21] to plan a vehicle trajectory in the presence of other vehicles and traffic lights. Piecewise linear constraints related to traffic light signals and other vehicles are constructed, and the trajectory is optimized for each piecewise linear segment of the constraints, as well as the trajectory for the whole horizon. The authors in [22] include TLS information as a soft constraint within the optimization problem, which is then solved by using deterministic dynamic programming (DDP). The problem can further benefit from the presence of vehicle-to-everything (V2X) (more specifically, vehicle-to-infrastructure (V2I)) communication [23].

In this way, traffic flow can be improved by the simultaneous optimization of both vehicle speed trajectories and traffic signals phases, as shown in [24], or by reserving a time slot for the vehicle to cross the signalized crossing by direct communication with the intelligent traffic light [25].

In this paper, the problem of autonomous vehicle control when approaching a signalized crossing is first addressed by proposing a linear MPC strategy, with the objective of maintaining reference velocity while obeying traffic lights and vehicle speed and acceleration constraints. Instead of applying the two-step approach to avoid the mixed-integer problem, the vehicle-position-dependent traffic light constraint is transformed into a time-dependent constraint by applying a linear time-varying (LTV) approach. In this approach, the constraint formulation is based on the optimization outcome from the previous sampling step, thus avoiding dual optimization or additional high-level strategies to ensure stopping at a red light. Different ways of reducing the dimensionality of the optimal control problem are considered, such as a shorter control horizon and a move-blocking scheme. To further reduce the dimensionality, an approach relying on the reference velocity profile mimicking a first-order lag element is proposed, in which only two control parameters (the lag time constant and the target velocity) are determined by a nonlinear model predictive control (NMPC) law, thus reducing the control horizon length to the minimum level of one sampling step. In this way, the computational load is significantly decreased when compared with a more conventional, full-horizon NMPC strategy, while the influence on control performance degradation is quite modest. The nonlinear MPC law is further improved in terms of computational efficiency, as well as stability in the general sense, by using multiple lag terms with a fixed-lag time constant, which results in multiple linear MPCs, or what is known in control theory as parallel MPC [26]. To avoid acceleration chattering due to linear lag model switching, a virtual-actuator-like low-pass filter is incorporated into the vehicle prediction model. The proposed NMPC strategy can be adjusted and applied in more general cases concerning nonlinear and stochastic process/prediction models, e.g., for solving the AV safe speed control problem while interacting with pedestrians when approaching an unsignalized crosswalk [27,28].

The emphasis of the presented study is on the design and comparative assessment of different characteristic MPC structures that can be applied in various automated driving tasks. To this extent, the scenario of AV approaching a signalized crosswalk is taken more as a case study for the AV MPC design, rather than a specific scenario for which a comprehensive control strategy is developed. The main contributions of the paper include: (i) a linear time-varying MPC approach that can handle traffic light constraints while avoiding the mixed-integer formulation; (ii) the design of an NMPC law with an ultimately reduced control input sequence based on a first-order lag profiled reference velocity; (iii) computationally efficient reduction of the NMPC law into a smooth, parallel MPC strategy, and (iv) comparative analysis of linear, nonlinear, and parallel MPC strategies with respect to control performance and computational time criteria.

The remainder of the paper is organized as follows. The vehicle modeling and control methods are presented in Sections 2 and 3. In addition to the vehicle model, Section 2 describes the considered autonomous vehicle control scenario and related constraints. Section 3 presents different model predictive control design methods for the specified automated driving scenario, which include linear, nonlinear, and parallel model predictive control. Section 4 presents simulation results and related comparative analysis of the MPC strategies proposed in Section 3. A discussion of results, including their implications and possible future extensions, is contained in Section 5. Concluding remarks are given in Section 6.

## 2. Vehicle and Scenario Model

The considered control scenario (Figure 1) concerns an autonomous vehicle with the initial position $s_0 = 0$ and the initial velocity $v_{0,}$ approaching a traffic light crossing positioned at the distance $L_0$ from the initial vehicle position. The objective is to control the vehicle velocity $v$ in an optimal manner, i.e., as close to the reference velocity $v_R$ (e.g., equal

to $v_0$), while maintaining comfortable acceleration $a$ and stopping safely at the red traffic light. The traffic light state $S$ is assumed to be deterministic and known in advance, where $S = 1$ and $S = 0$ indicate red light and green light, respectively.

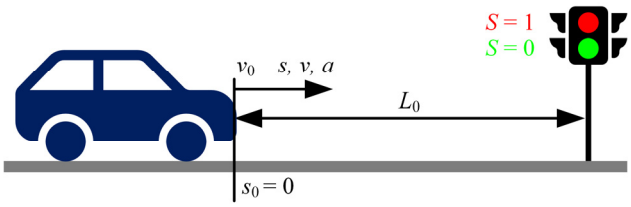


**Figure 1.** Traffic light crossing scenario schematic.

The vehicle is modeled by using the following point-mass state-space model

$$\begin{bmatrix} \dot{s} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} s \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} a \tag{1}$$

where $s$ and $v$ are the vehicle position and velocity state variables, respectively, while $a$ is the vehicle acceleration control input. Note that Model (1) assumes that the acceleration control input is realized through a high-bandwidth powertrain (e.g., that of an electric vehicle), and that the nonlinear rolling resistance and aerodynamic drag terms [29] are compensated for through either feedforward terms or a high-bandwidth vehicle acceleration inner control loop. As needed, the model can be extended to account for powertrain or acceleration loop lag dynamics, as demonstrated in Section 3 as an example of including a virtual actuator submodel. Discretizing Model (1) using Z-transform and zero-order hold element with sampling time $T_s$ yields the following discrete-time model:

$$\begin{bmatrix} s(k+1) \\ v(k+1) \end{bmatrix} = \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix} \begin{bmatrix} s(k) \\ v(k) \end{bmatrix} + \begin{bmatrix} \frac{1}{2}T_s^2 \\ T_s \end{bmatrix} a(k). \tag{2}$$

The vehicle velocity and acceleration are constrained through inequalities

$$\begin{aligned} a_{\min} \;&\leq\; a \;\leq\; a_{\max}, \\ v_{\min} \;&\leq\; v \;\leq\; v_{\max}, \end{aligned} \tag{3}$$

which are applied in both simulation and prediction models. Additionally, the vehicle should stop at a red traffic light, which yields the following conditional constraint on vehicle position depending on the traffic light state time profile $S(k)$:

$$s(k) \leq \begin{cases} L_0, & \text{for } S(k) = 1 \text{ and } s(k-1) \leq L_0 \\ \infty, & \text{otherwise} \end{cases} \tag{4}$$

The length of the AV-anticipated traffic light preview, i.e., the prediction horizon, can be limited to a fixed, relatively narrow value, or a full preview could be used, e.g., the one which stretches at least up to the next green traffic light state. It is beneficial for the control algorithm if the fixed preview (related to receding horizon length) is long enough for the vehicle to reach the first traffic light position, assuming a constant initial velocity, which gives

$$t_a \geq \frac{L_0}{v_0}. \tag{5a}$$

Furthermore, the preview should also be long enough for the vehicle to be able to timely come to a full stop when needed, under the assumption of the worst-case scenario of the vehicle having the maximum allowed speed, which gives

$$t_b \geq \frac{v_{\max}}{|a_{\min}|}. \tag{5b}$$

To improve the quality of vehicle speed optimization, the preview should also be long enough to cover the traffic light change from the current state:

$$t_c \geq t_{tls}, \tag{5c}$$

where $t_{tls}$ is the remaining duration of the current traffic light state. Finally, the prediction horizon length is conservatively determined when initializing the controller (at $t = 0$) as the highest value of the above-defined three time limits

$$t_p = \max(t_a, t_b, t_c) \tag{5d}$$

## 3. Design of Model Predictive Controllers

For the scenario described in Section 2, various model predictive control (MPC) laws (linear, nonlinear, and parallel) are applied to control the vehicle in an optimal, preemptive, and closed-loop manner. The MPC algorithms solve a constrained finite-horizon optimal control problem at each control time step $k$ to obtain an optimal acceleration sequence and apply the first element of the sequence to the vehicle. The feedback control is realized by repeating the optimization algorithm on the receding horizon with updated measured actual states (vehicle position and velocity). Generally, to formulate the MPC optimal control problem, three building blocks are required: plant dynamics prediction model, cost function, and constraints.

### *3.1. Linear Model Predictive Control*

#### 3.1.1. Control Law Design

Linear MPC considers a linear prediction model, linear constraints, and a quadratic cost function, which results in a quadratic program (QP) that could be efficiently solved online. At a given control time step $k$, MPC minimizes a cost function $J_{MPC}$ on the prediction horizon $h = 0, \ldots, N_p - 1$ by optimizing the sequence of control inputs $u(h \mid k)$ over the control horizon $h = 0, \ldots, N_c \leq N_p$ [30]. The first input, $u(0 \mid k)$, is applied to the vehicle as the actual control input. Note that the prediction horizon length is determined as $N_p = \text{int}(t_p/T_s)$, where $t_p$ is given by Equation (5d).

In the considered scenario, the linear MPC law is designed with the aim of minimizing the vehicle velocity regulation squared error and the squared acceleration as a comfort index, which results in the following quadratic cost function:

$$\min_{a(h|k)} J_{MPC} = \sum_{h=0}^{N_p-1} q_v (v(h|k) - v_R)^2 + \sum_{h=0}^{N_c-1} q_a a^2(h|k) \tag{6}$$

where $v_R$ is the constant reference velocity (Section 2), and $q_v$ and $q_a$ are the velocity regulation and acceleration weighting coefficients, respectively. Alternative formulations can be considered, e.g., those that employ the minimum time or maximum distance traveled criterion instead of the speed regulation criterion. The cost function (6) is subject to the equality constraints set by the discrete-time plant dynamics model (2):

$$\underbrace{\begin{bmatrix} s(h+1|k) \\ v(h+1|k) \end{bmatrix}}_{\mathbf{x}(h+1|k)} = \underbrace{\begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} s(h|k) \\ v(h|k) \end{bmatrix}}_{\mathbf{x}(h|k)} + \underbrace{\begin{bmatrix} \frac{1}{2}T_s^2 \\ T_s \end{bmatrix}}_{\mathbf{B}} \underbrace{a(h|k)}_{u(h|k)}. \tag{7}$$

By following Equations (3) and (4), the following hard constraints are applied to acceleration, velocity, and position on the prediction horizon:

$$\begin{aligned} a_{\min}(h|k) \leq a(h|k) \leq a_{\max}(h|k) \\ v_{\min}(h|k) \leq v(h|k) \leq v_{\max}(h|k) \end{aligned} \tag{8}$$

$$s(h|k) \leq \begin{cases} L_0, & \text{for } S(h|k) = 1 \text{ and } s(h-1|k) \leq L_0 \\ \infty, & \text{otherwise} \end{cases} \tag{9}$$

Constraint (9) introduces the traffic light state preview in the optimal control problem. However, since the vehicle position $s$ is a state that is to be optimized, applying this constraint directly in the optimal control problem formulation would result in a logical implication and, thus, in a mixed-integer QP form. To avoid the logical implication within the optimal control problem, the traffic-light-related constraint for current optimization step $k$ is constructed by using the vehicle position trajectory predicted in the previous optimization step, i.e., $s(h\,|\,k-1)$ ($h = 0, \ldots, N_p - 1$), which is similar to the linear time-varying approach in model predictive control. The overall constraint construction algorithm is given in Appendix A, whereas Figure 2 illustrates the main idea of the algorithm. In the first step of optimization ($k = 0$, Figure 2a), the vehicle is predicting that it will cross the traffic light during the first red-light state. That information is utilized in the next optimization step ($k = 1$, Figure 2b), in which, based on the predicted vehicle position (black line) in step $k = 0$, the traffic light position constraint based on the original traffic light state prediction $S$ (dashed blue line) is modified and becomes active for the first red-light state (green dotted line), thus ensuring that the vehicle crosses only after the first red-light phase is over. The second red-light phase position constraint is not active, since it is anticipated that the vehicle is going to cross before the start of that phase. Since this procedure relies on the previously predicted vehicle position instead of the current vehicle position (cf. Equation (9)), the final traffic light position-related constraint becomes time-dependent only and is formulated as

$$s(h|k) \leq \begin{cases} L_0, & \text{for } S_{mod}(h|k) = 1 \\ \infty, & \text{otherwise} \end{cases} \tag{10}$$

where $S_{mod}(h\,|\,k)$ is the modified traffic light state prediction, which is formally defined in Appendix A.

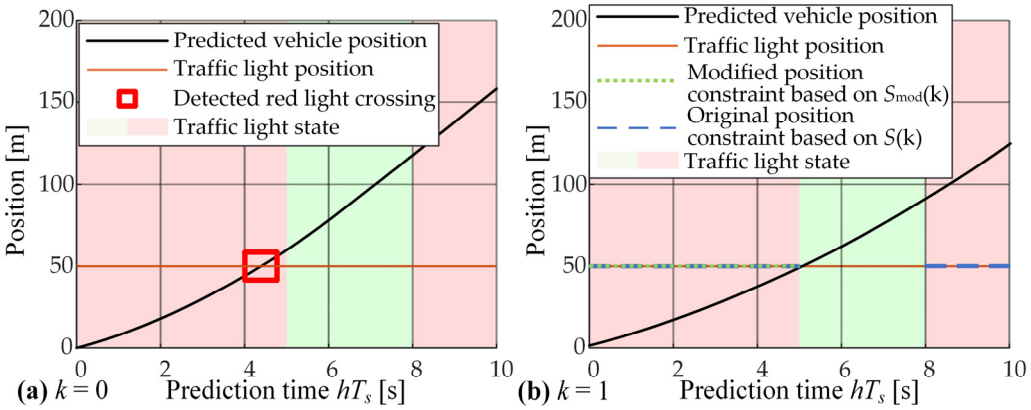


**Figure 2.** Illustration of construction of time-dependent only traffic light position constraint in current time step (**b**) based on vehicle position trajectory predicted in previous step (**a**).

After the constraint reconstruction, the MPC optimal control problem, given by Equations (6)–(8) and (10), can be rewritten into the standard QP form:

$$\begin{aligned} &\min_{\xi} \mathbf{J}_{QP} = \tfrac{1}{2}\xi^{\mathrm{T}}\mathbf{H}_{QP}\xi + \mathbf{f}_{QP}^{T}\xi + \mathbf{S}_{QP}, \\ &\text{subject to : } \mathbf{A}_{ineqQP}\xi \leq \mathbf{b}_{ineqQP}, \end{aligned} \tag{11}$$

where the vector $\xi$ contains a sequence of control inputs $\xi = [u(0\,|\,k), \ldots, u(N_c\,|\,k)]^{\mathrm{T}}$, while the cost-function-related Hessian symmetric positive-definite matrix $\mathbf{H}_{QP}$, linear cost vector $\mathbf{f}_{QP}$, constant term $\mathbf{S}_{QP}$, and inequality constraint-related matrix $\mathbf{A}_{ineqQP}$ and vector $\mathbf{b}_{ineqQP}$ are derived from Equations (6) to (8) and (10). Herein, the optimal control problem is

automatically transformed into the QP form (11) within MATLAB by using a dedicated toolbox CasADi [31].

The QP problem (11) can be solved by different quadratic program solvers, such as the *quadprog* solver available within MATLAB or *mpcInteriorPointSolver* from MATLAB's MPC Toolbox. Here, the open-source solver qpOASES is used to solve the automatically generated QP [32].

#### 3.1.2. Optimal Control Problem Size Reduction

The above-described nominal MPC case corresponds to full optimization of control input $u(h\,|\,k)$, with $h = 0, \ldots, N_c - 1 = N_p - 1$ (see Figure 3a). To reduce the size of the optimal control problem and, therefore, improve the computational efficiency, two different strategies are considered below.

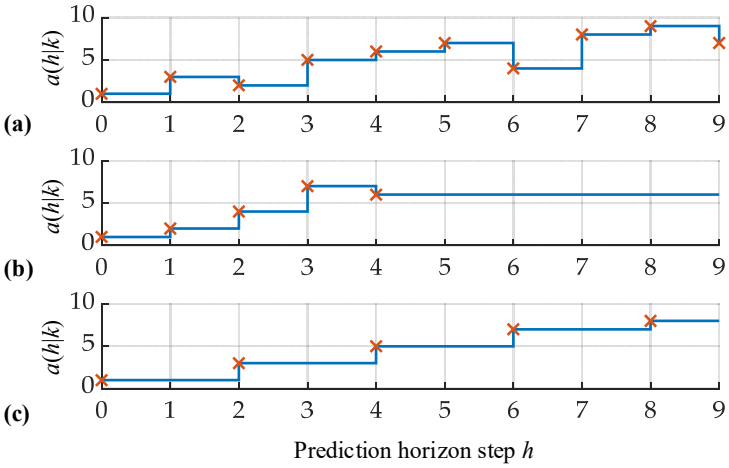


**Figure 3.** Illustration of full control horizon (**a**), limited control horizon (**b**), and move-blocking strategies (**c**).

The first strategy relates to the typical option of reducing the control horizon, i.e., setting $N_c < N_p$, and holding the last control input $u(N_c - 1\,|\,k)$ until the end of the prediction horizon (Figure 3b):

$$a(h|k) = \begin{cases} u(h|k), & \text{for } h < N_c \\ u(N_c - 1|k), & \text{otherwise} \end{cases} \tag{12}$$

A drawback of this strategy is the potential infeasibility of the QP for very short control horizons $N_c << N_p$, because the optimizer may not have enough freedom to shape the acceleration profile to stop the vehicle before the red-state traffic light. This is illustrated in Appendix B for the case of $N_c$ reduced to the lowest feasible value, which allows the vehicle to fully stop before the red light with maximum acceleration. Note that the lower limit of feasible control horizon $N_c$ is dependent on the scenario conditions, such as the initial position and velocity, and the acceleration limits, as well.

The second option is the so-called move-blocking strategy [33], in which the control input is held constant over several consecutive prediction horizon steps, which is similar to the downsampling of the control input rate (see Figure 3c). Note that the block width can be fixed or varying, resulting in uniformly or unequally distributed blocks. Herein, a uniform distribution is opted for, and the number of blocks $N_b$ is chosen as a multiple of prediction horizon $N_p$, which gives

$$a(h|k) = \begin{cases} u(h|k), & \text{if } \mathrm{mod}(h, N_b) = 0 \\ u(h-1|k), & \text{otherwise} \end{cases} \tag{13}$$

resulting in $N_p/N_b$ unique control inputs, where mod(.) denotes the modulo function. This formulation maintains a feasible QP in all scenarios, as it allows the optimizer more

flexibility in shaping the acceleration and velocity over the whole prediction horizon when compared to the case of limiting the control horizon length, whereas the drawback is the lower resolution of the control input.

### *3.2. Nonlinear Model Predictive Control*

Figure 4 illustrates a novel MPC approach, in which two parameters of the prediction horizon velocity profile are optimized instead of the acceleration at each prediction time step. The initial idea would be to optimize the target velocity $v_F$ and the initial acceleration $a_0 = (v_F - v_0)/t_F$, which defines the piecewise linear velocity profile shown by dashed lines in Figure 4. To avoid the discontinuity of the piecewise linear profile in its breakpoint, the velocity trajectory is approximated by a first-order lag term response, which is defined by the target velocity $v_F$ and the time constant $T_F$:

$$T_F\dot{v}(t) + v(t) = v_F \tag{14}$$

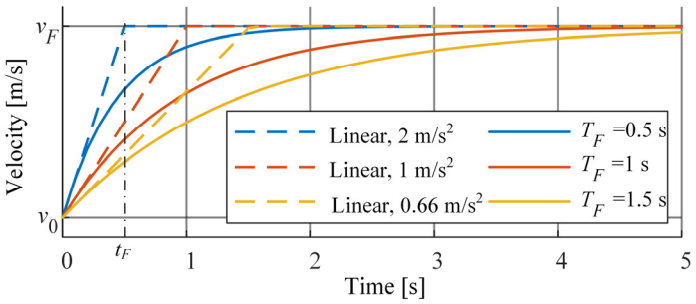


**Figure 4.** Illustration of nonlinear MPC velocity trajectory.

Note that the time constant $T_F$ directly determines the initial and at the same time maximum acceleration, which is equal to $a_0 = (v_F - v(0))/T_F$ (see Figure 4).

To incorporate the specific velocity trajectory defined by Equation (14) in the MPC formulation, Model (1) is modified into the following form:

$$\begin{aligned} \dot{s} &= v, \\ \dot{v} &= -\tfrac{1}{T_F}v + \tfrac{1}{T_F}v_F = -u_2 v + u_1 u_2, \end{aligned} \tag{15}$$

where the control inputs are the target velocity $u_1 = v_F$ and the time constant inverse, i.e., the bandwidth $u_2 = 1/T_F$. Due to the multiplications between control inputs and states, the model given by Equation (15) is nonlinear. In addition, the acceleration constraint (8), where $a$ is equal to $\dot{v}$ defined in Equation (15), is also nonlinear. For such a nonlinear problem, it is generally convenient to use nonlinear MPC (NMPC). To obtain the discrete-time model, the state-space system (15) is discretized by using the forward Euler method. As an alternative, the more accurate and computationally less efficient fourth-order Runge–Kutta method can be applied [34].

The cost function (6) is expanded with the control input rate penalization to achieve oscillation-free commands. In order to keep $u_1$ and $u_2$ constant on the prediction horizon for a favorable computational efficiency, the control horizon is set to the minimum value $N_c = 1$ (note that, in this case, the predicted velocity response corresponds to the one shown in Figure 4). The final NMPC cost function is

$$\begin{aligned} \min_{u_2,u_1} J = \sum_{h=0}^{N_p-1} \Big( & q_v(v_R - v(h|k))^2 + q_a(-u_2(k)v(h|k) + u_1(k)u_2(k))^2 \Big) \\ & + r_1\Delta u_1^2 + r_2\Delta u_2^2, \end{aligned} \tag{16}$$

where $\Delta u_1 = u_1(k) - u_1(k-1)$ and $\Delta u_2 = u_2(k) - u_2(k-1)$. The cost function is subject to Constraints (8) and (10), and the following constraints on control inputs:

$$v_{\min} \leq u_1(k) \leq v_{\max} \tag{17}$$

$$1/T_{Fmax} \leq u_2(h|k) \leq 1/T_{Fmin} \tag{18}$$

In order to solve the optimal control problem given by the discrete-time version of Equations (15)–(18), a nonlinear program is constructed by applying the direct multiple shooting method [34]. Within MATLAB, this is again performed automatically by using the CasADi toolbox, and the nonlinear program is solved by using the interior-point method solver IPOPT [35].

### *3.3. Parallel Model Predictive Control*

The nonlinear process model (15) transforms into a linear model by fixing the time constant $T_F$. For the fixed inverse of time constant $T_F$, the control input $u_2$ becomes a parameter $\kappa = 1/T_F$, which gives the linear model

$$\begin{aligned} \dot{s} &= v \\ \dot{v} &= -\kappa v + \kappa v_F \end{aligned} \tag{19}$$

The corresponding cost function (16) becomes (cf. Equation (16)):

$$\min_{u_1} J = \sum_{h=0}^{N_p-1} \left( q_v (v_R - v(h|k))^2 + q_a \kappa (-v(h|k) + u_1(k))^2 \right) + r_1 \Delta u_1^2, \tag{20}$$

which is also subject to Constraints (8), (10), and (17). Each individual linear MPC optimal control problem is transformed into QP by using the same procedure as explained in Section 3.1. Parallel MPC (PMPC) that mimics NMPC is realized by formulating $M$ linear process models with different predefined time constants $\kappa$, and applying $M$ MPC laws in parallel at each sampling instant $k$, which is illustrated in Figure 5. Once all the MPC solutions are available, the final PMPC output, i.e., the control input $a(k)$ is adopted from MPC that gives the lowest cost function (20) (see equation given in Figure 5).

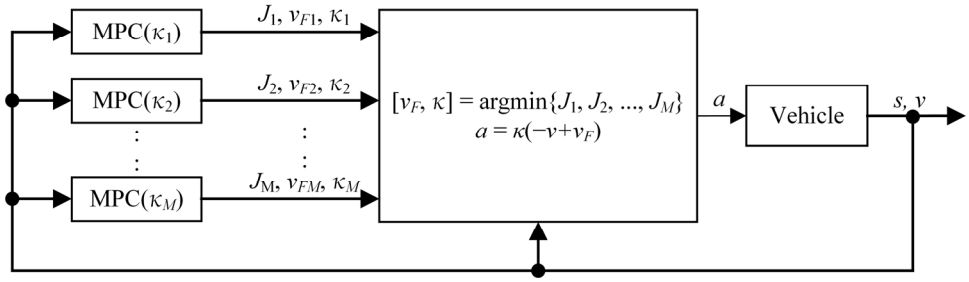


**Figure 5.** Block diagram of parallel MPC.

Due to the finite number of ($M$) linear MPCs, the parameter $\kappa = 1/T_F$ changes in a stepwise manner when switching individual MPC laws. This would result in a stepwise change of acceleration command $a_R = -\kappa(v + v_F)$, thus causing peaks of vehicle jerk and affecting the driving comfort. To mitigate the discomfort, one may increase the number $M$ of parallel MPCs at the expense of increased computational effort. Alternatively, the PMPC command can be filtered with an acceleration command filter, which results in the PMPCf strategy shown in Figure 6. The filter may be regarded as a virtual actuator, which

should be incorporated into the process model. By assuming the first-order lag-type virtual actuator, the extended process model is formulated as (cf. Equation (19)):

$$\begin{aligned} \dot{s} &= v \\ \dot{v} &= x_f \\ \dot{x}_f &= -\kappa_f x_f + \kappa_f \left( -\kappa v + \kappa v_f \right) \end{aligned} \tag{21}$$

where $x_f$ is the virtual actuator state that is equal to the actual acceleration $a$ as a response to the acceleration command $a_R = -\kappa(v + v_F)$, and $\kappa_f = 1/T_f$ is the virtual actuator bandwidth (Figure 6). Once the process model (21) is defined, PMPCf is designed by using the same procedure as described in Figure 5 and the comments above.

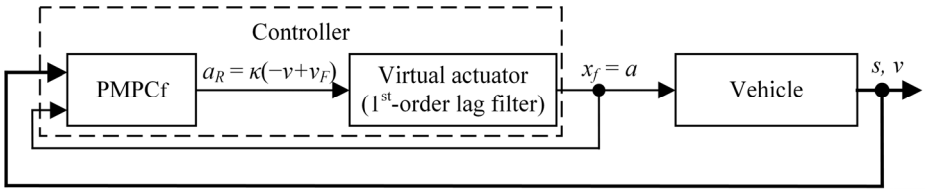


**Figure 6.** Block diagram of parallel MPC with virtual actuator.

## 4. Simulation Results

### *4.1. Simulation Setup and Performance Metrics*

This section presents a comparative simulation analysis of the model predictive strategies proposed in Section 3. In the considered scenario, the vehicle should maintain typical city driving velocity $v_R = 15$ m/s. The initial distance between the vehicle and the traffic light is set to $L_0 = 150$ m, and the traffic light period is set to 20 s with green- and red-light periods of 8 and 12 s, respectively. The vehicle should maintain the acceleration within the bounds of $a_{\max} = -a_{\min} = 5$ m/s$^2$, whereas the velocity should stay between $v_{\min} = 0$ and $v_{\max} = 20$ m/s. The sampling time is set to $T_s = 0.1$ s (if not stated otherwise). Full traffic light preview is utilized, i.e., $N_p = 200$ steps (unless otherwise stated), and in the case of linear MPC, $N_c = N_p$ is set. Nominally, the cost function weights are chosen to $q_v = 10$, $q_a = 5$, $r_1 = 0.1$, and $r_2 = 0.1$. Unless stated otherwise, PMPC and PMPCf are designed with 10 linear MPCs ($M = 10$), and the move-blocking MPC is designed with $N_b = N_p/10 = 20$ uniformly distributed blocks.

The overall performance of the control systems is compared in terms of the vehicle velocity RMS regulation error

$$v_{\text{RMS}} = \sqrt{\frac{1}{t_{sim}} \int_0^{t_{sim}} (v_R - v)^2 dt}, \tag{22}$$

RMS acceleration (discomfort index)

$$a_{\text{RMS}} = \sqrt{\frac{1}{t_{sim}} \int_0^{t_{sim}} a^2 dt}, \tag{23}$$

traveled distance:

$$s_{\max} = s(t_{sim}) \tag{24}$$

and computational time $t_{exe}$ evaluated on a personal computer based on Intel® i7 central processing unit operating at 2.8 GHz.

### *4.2. Linear MPC*

Figure 7 illustrates the effect of reducing the prediction horizon length $N_p$ from the nominal value of 200 to 50. In the case of a limited horizon of only 50 steps (i.e., 5 s), the vehicle swiftly accelerates to the reference velocity of 15 m/s (Figure 7b) while respecting

the acceleration limit of 5 m/s$^2$ (Figure 7c), maintains this velocity, and then slows down and almost stops before the traffic light (Figure 7a). This is because the prediction horizon is not long enough to plan for the upcoming red light. Once the light turns green, the vehicle accelerates again toward the reference velocity (Figure 7b) and continues driving at that velocity. For the case of a full horizon (20 s), the vehicle maintains a constant velocity lower than the reference value, which allows it to safely approach and then cross the traffic light at $t$ = 20 s. By doing so, the vehicle turns out to have a higher velocity when crossing the traffic light and achieves a somewhat higher final position than in the limited preview case. More importantly, the acceleration is smoother, and, thus, the comfort level is higher for $N_p$ = 200 than for $N_p$ = 50. While the longer prediction horizon allows for better traffic light anticipation and overall performance, it results in significantly higher execution times than the limited one, as illustrated in Figure 7d.

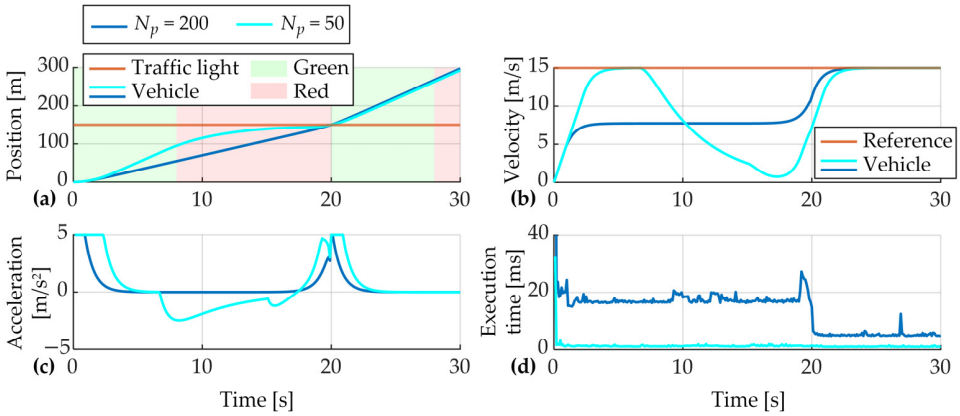


**Figure 7.** Comparative responses position (**a**), velocity (**b**), acceleration (**c**) and execution time (**d**) of linear MPC systems for prediction horizon lengths $N_p$ = 200 and $N_p$ = 15.

Figure 8 compares the responses corresponding to different initial velocities: $v_0$ = 0 and $v_0 = v_R$ = 15 m/s. In both cases, the nominal value of prediction horizon length is used ($N_p$ = 200), and the resulting performance is similar, i.e., the vehicle adjusts its velocity to a similar value that allows it to safely cross the traffic light right when it turns green ($t$ = 20 s). In the case of a high initial velocity ($v_0$ = 15 m/s), the vehicle immediately starts slowing down with the maximum allowed deceleration (5 m/s$^2$) to a constant velocity. Similarly, for zero initial velocity, the vehicle immediately starts speeding up with the maximum acceleration (5 m/s$^2$) to a similar constant velocity. After crossing the traffic light, the vehicle accelerates to the reference velocity in both cases. The MPC code execution time is comparable to that of Figure 7 for the same (full) prediction horizon length $N_p$ = 200 (Figure 8d).

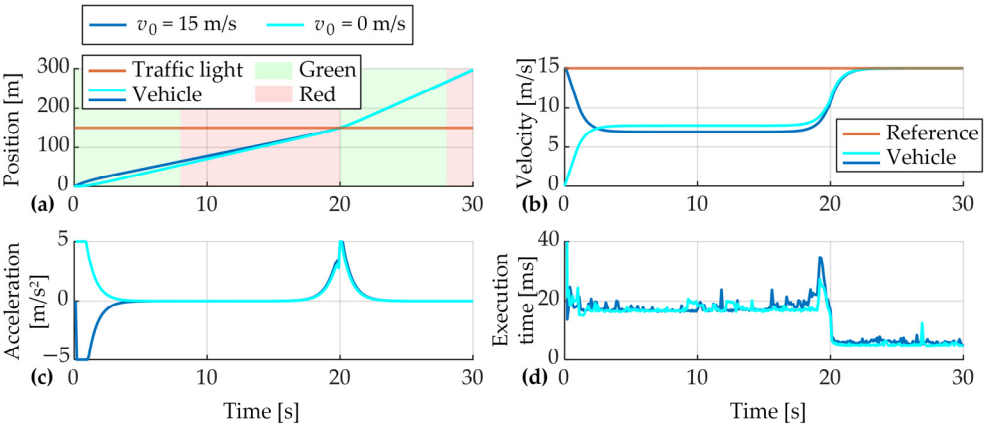


**Figure 8.** Comparative responses position (**a**), velocity (**b**), acceleration (**c**) and execution time (**d**) of linear MPC systems for initial vehicle velocities $v_0$ = 0 and $v_0 = v_R$ = 15 m/s.

The comparative performance of nominal (full) MPC and move-blocking MPC (MPC-MB) is shown in Figure 9. As expected, the move-blocking strategy significantly reduces the code execution time compared to the nominal MPC (Figure 9d), while the performance is only slightly worse. The performance degradation is due to holding the control input constant over several consecutive prediction steps (see illustration in Figure 3), which reduces the MPC's flexibility in shaping the control input.

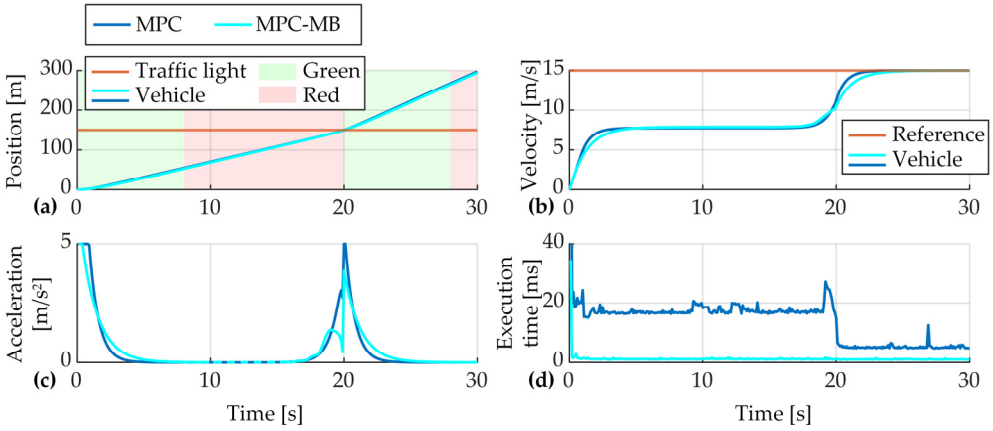


**Figure 9.** Comparative responses position (**a**), velocity (**b**), acceleration (**c**) and execution time (**d**) of nominal and move-blocking (MB) linear MPC systems.

The effect of cost function tuning is illustrated in Figure 10. Increasing the acceleration penalization results in decreased acceleration peaks (Figure 10c), i.e., better comfort. MPC achieves this by applying lower acceleration values over a longer period. This means that the vehicle velocity will reach the steady velocity more slowly, (Figure 10b), thus resulting in somewhat worse performance.

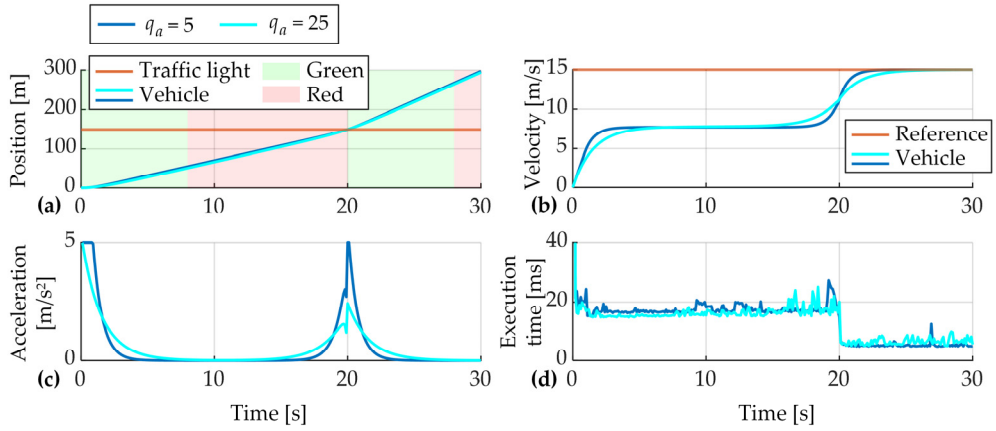


**Figure 10.** Illustration of influence of acceleration weighting coefficient $q_a$ on position (**a**), velocity (**b**), acceleration (**c**) and execution time (**d**).

### *4.3. Nonlinear MPC*

Figure 11 illustrates the influence of the nonlinear process discretization method on the NMPC performance. Applying the Euler forward method and the fourth-order Runge–Kutta method gives a similar system response. Based on 100 simulations with similar initial conditions, the use of the Euler method results in a reduction of the average code execution time by 24%.

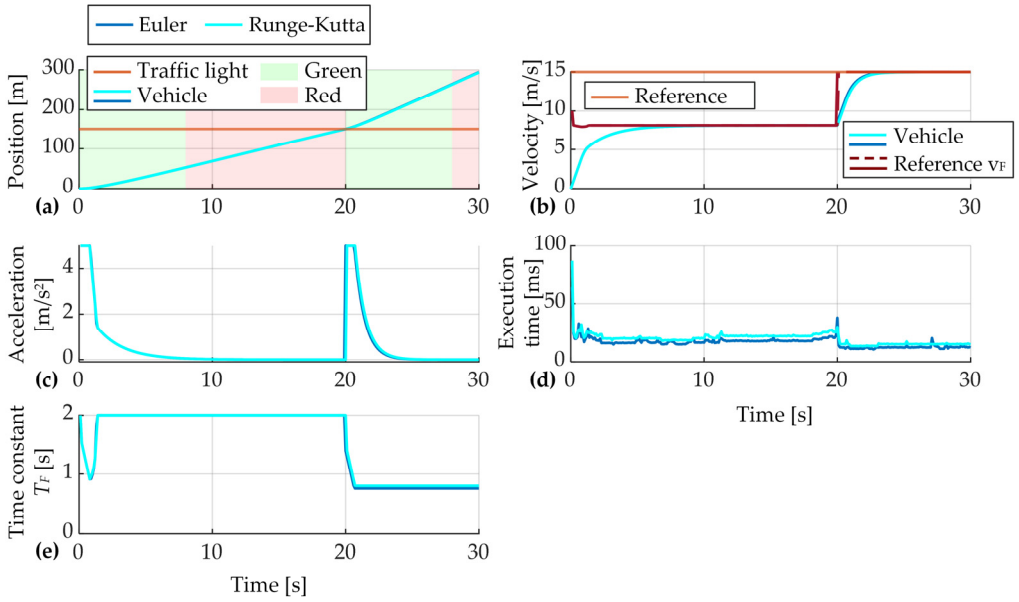


**Figure 11.** NMPC system responses for different time-discretization methods applied. Subfigures show position (**a**), velocity (**b**), acceleration (**c**), execution time (**d**) and time constant control input (**e**).

The influence of the sampling time selection in the Euler discretization method is illustrated in Figure 12. Increasing the sampling time from 100 ms to 200 ms, and accordingly reducing $N_p$ from 200 to 100, marginally impacts the overall system performance. However, by selecting the higher sampling time, the code execution time is reduced by 34% on average.

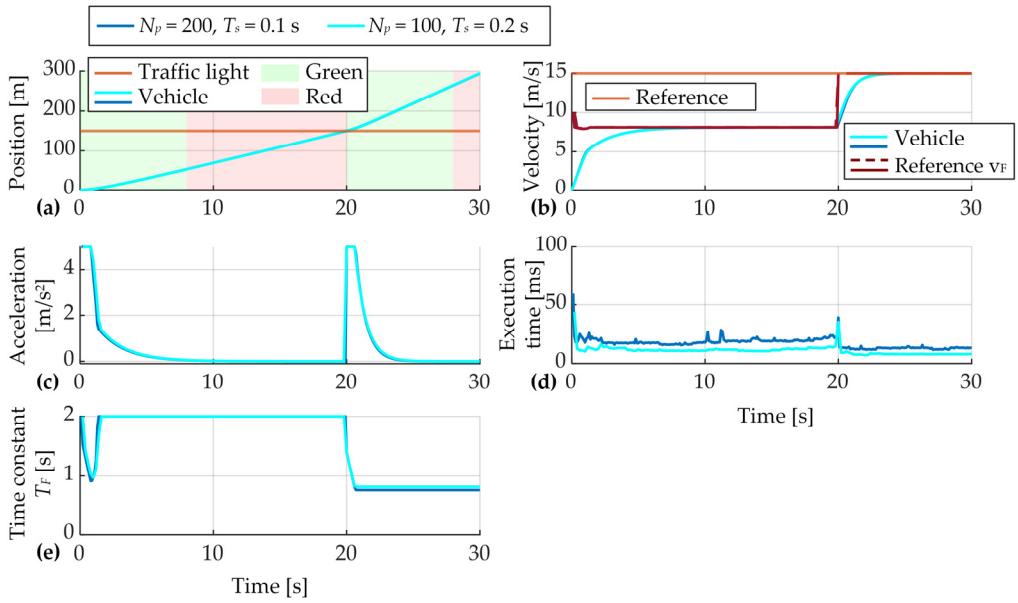


**Figure 12.** NMPC system responses for different sampling time values. Subfigures show position (**a**), velocity (**b**), acceleration (**c**), execution time (**d**) and time constant control input (**e**).

The comparison of the linear and nonlinear MPC systems is supported through responses shown in Figure 13. The linear MPC opts for slightly lower approaching velocity during the red-light period (Figure 13b), which allows it to start accelerating before the traffic light turns green (Figure 13c). In this way, MPC can cross the traffic light with a higher velocity and consequently travel further than NMPC. MPC can achieve this since it has the flexibility to fully shape acceleration and velocity over the whole prediction horizon,

whereas NMPC is limited to optimizing only two control parameters (the target velocity $v_F$ and time constant $T_F$, Figure 13b,e). This effectively means that, in the NMPC case, accelerating before $t$ = 20 s is not possible. The NMPC execution time is comparable to that of MPC, as the burden of applying a more complex nonlinear system optimization solver is apparently compensated for by the bare minimum number of control profile parameters.

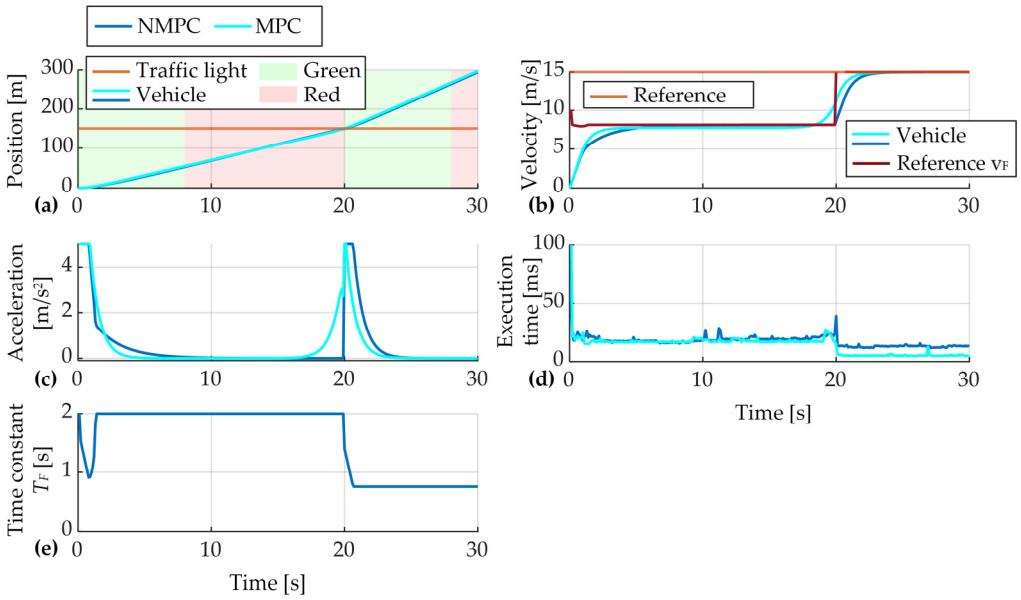


**Figure 13.** Comparative responses of position (**a**), velocity (**b**), acceleration (**c**), execution time (**d**) and time constant input (**e**) of MPC and NMPC systems.

In the NMPC case, the optimizer sets the target velocity $v_F$ to similar values as in the MPC case in the first half of the response. It then relies on the time constant $T_F$ to shape the acceleration profile, namely it achieves higher acceleration by reducing $T_F$, with a similar effect on the acceleration response as in the case of the more flexible MPC.

### *4.4. Parallel MPC*

Figure 14 shows comparative responses of the NMPC, PMPC, and PMPCf systems. PMPC consists of $M$ = 10 linear MPCs, with $\kappa$ values logarithmically spaced between 0.5 $s^{-1}$ ($T_{F\max}$ = 2 s) and 5 $s^{-1}$ ($T_{F\min}$ = 0.2 s). The overall behavior of the NMPC and PMPC systems is similar, with the difference that the time constant $T_F(t)$ commanded by PMPC has a stepwise shape (Figure 14e), thus resulting in an acceleration chattering effect (Figure 14c). However, the PMPC execution takes approximately 9.8 ms on average, which is faster than the NMPC execution time of 16.3 ms. Equally important, the execution time of PMPC is much more consistent, which facilitates the implementation and real-time application (Figure 14d).

Accounting for the virtual actuator dynamics through the design of PMPCf makes the acceleration response smooth for improved comfort (Figure 14c). Including an additional state variable in the PMPCf prediction model (cf. Equations (19) and (21)) has a negligible effect on the execution time (Figure 14d).

### *4.5. Comparison of Performance Metrics*

Table 1 presents a comparison of the performance metrics of different MPC strategies. The nominal (linear) MPC strategy is considered a performance benchmark, as it gives the minimum value of the cost function $J$ given by Equation (6), maximum traveled distance, and the lowest RMS velocity tracking error due to its highest flexibility in shaping the acceleration profile. The move-blocking MPC strategy is characterized by the best comfort and the lowest execution time. However, this strategy could be prone to feasibility issues. Although NMPC is

the least flexible in terms of shaping the acceleration profile, its performance is still comparable to other, more flexible strategies. PMPC also provides comparable results to the nominal MPC, where the increase in parallel MPC components from $M = 10$ to $M = 20$ results in proportional increases in the execution time, without tangible performance improvement. On the other hand, for a comparable level of overall performance, the execution time of PMPCf is considerably lower than that of the nominal case, particularly when the number of parallel MPC components is reduced from $M = 10$ to $M = 5$.

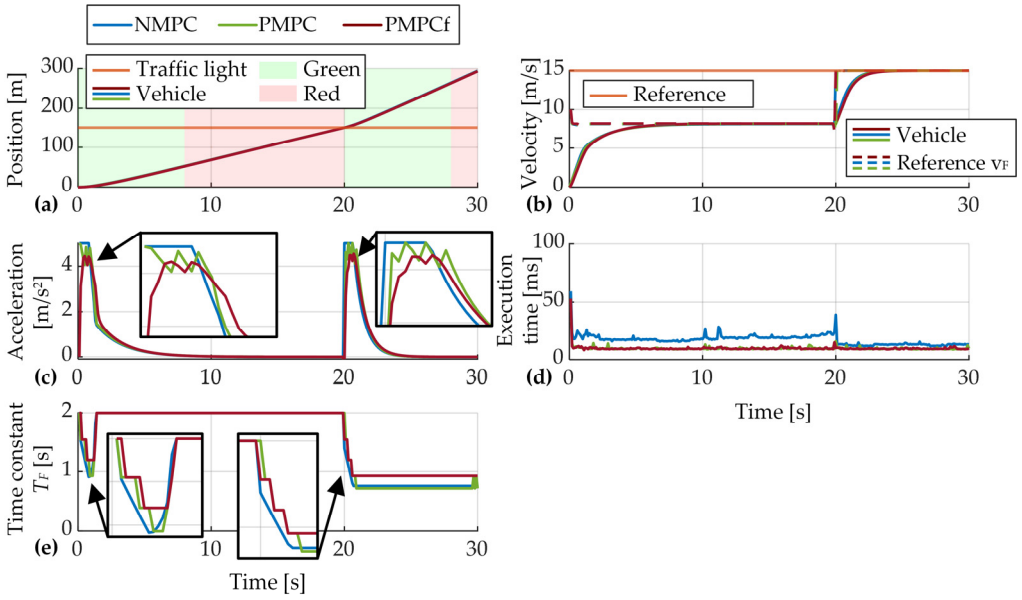


**Figure 14.** Comparative responses of position (**a**), velocity (**b**), acceleration (**c**), execution time (**d**) and time constant input (**e**) of NMPC, PMPC, and PMPCf systems.

**Table 1.** Summary of performance metrics for presented strategies.

| MPC | Overall Cost * $J \times 10^5$ | Discomfort Index $a_{RMS}$ (m/s$^2$) | Velocity Regulation RMS Error $v_{RMS}$ (m/s) | Traveled Distance $s_{max}$ (m) | Execution Time $t_{exe}$ (ms) |
|---|---|---|---|---|---|
| Linear MPC | 1.2007 | 1.2661 | 5.1137 | 295.8402 | 13.9 |
| MPC-MB | 1.2081 (+0.6%) | 1.0956 (−13.5%) | 5.2070 (+1.8%) | 293.0414 (−0.9%) | 1.2 (−91.4%) |
| NMPC | 1.2210 (+1.7%) | 1.3437 (+6.1%) | 5.2136 (+2.0%) | 292.8537 (−1.0%) | 16.3 (+17.3%) |
| PMPC ($M = 10$) | 1.2286 (+2.3%) | 1.3132 (+3.7%) | 5.2438 (+2.5%) | 291.9483 (−1.3%) | 9.8 (−29.5%) |
| PMPC ($M = 20$) | 1.2272 (+2.2%) | 1.3281 (+5.0%) | 5.2413 (+2.5%) | 292.0224 (−1.3%) | 19.1 (+37.4%) |
| PMPCf ($M = 10$) | 1.2304 (+2.5%) | 1.2009 (−5.1%) | 5.2512 (+2.7%) | 291.7150 (−1.4%) | 9.8 (−29.5%) |
| PMPCf ($M = 5$) | 1.2365 (+3.0%) | 1.1424 (−9.8%) | 5.2851 (+3.4%) | 290.6967 (−1.7%) | 5.0 (−64.0%) |

* Overall cost is calculated by using Equation (6).

## 5. Discussion

The presented study has focused on the design and comparative assessment of several practical MPC strategies. The strategies have been proven to be viable through simulation verification for a typical AV case study. However, before implementing the designed MPC

strategies in the considered or a broader AV control scenario, there are several, mostly practical, considerations that have to be accounted for. Firstly, in real-world applications, the traffic light state preview will contain some uncertainty. Therefore, the proposed algorithms should be tested against this uncertainty, and they can be potentially modified or extended for improved performance in the more complex, stochastic environment. One of the issues that could arise is related to QP problem infeasibility when a vehicle is close to the traffic light and its speed is not adapted soon enough due to the presence of uncertainties. To prevent such cases, traffic light position-related constraints should be relaxed by introducing a buffer zone, e.g., a 1–2 m long zone, before the traffic light, and a slack variable should be applied to the position and acceleration constraints to maintain the QP feasibility while stopping at a safe distance. Similarly, the proposed algorithms should be tested and potentially modified for the scenarios of multiple traffic lights and multiple vehicles.

Next, the presented MPC approaches can accommodate a more detailed vehicle model, which would consider more detailed powertrain lag dynamics and would include feedforward or feedback-based compensation of aerodynamic drag and rolling resistance. The extended vehicle model could also include a power consumption submodel. In this case, the MPC cost function could be extended with an energy/fuel consumption term to directly account for efficiency (e.g., eco-driving preference of the driver). Similarly, the cost function can further be reformulated to account for different performance criteria. For instance, instead of minimizing the vehicle velocity regulation error, one can maximize the distance traveled. In that case, the vehicle would generally be encouraged to accelerate to pass the traffic light crossing before the light turns red. The developed MPC framework should readily be adjusted to allow for such reformulations of the cost function.

Additionally, the robustness of the proposed strategies to unmodeled powertrain and sensor dynamics should be verified. Besides the robustness analysis, a more detailed computational efficiency analysis should be carried out to determine appropriate hardware requirements for the in-vehicle implementation of the control algorithms and/or establish the tuning trade-off between traffic light preview duration and real-time capability.

Finally, although the considered case study has assumed autonomous driving, the proposed MPC approaches are not limited to those tasks only. They can be extended and integrated into existing advanced driving assistance systems, such as adaptive cruise control, where the speed of the vehicle could be reduced based on the traffic light preview, and automatic emergency braking systems, which would automatically stop the vehicle if the driver does not timely react to red traffic light.

## 6. Conclusions

The presented comparative simulation results have indicated that all the three predictive control approaches (MPC, NMPC, and PMPC) can successfully be applied to control an AV in the considered scenario of approaching a traffic light. The main advantage of the proposed NMPC law is that it accommodates a nonlinear process model without compromising computational efficiency. This is achieved by describing the vehicle velocity profile by only a pair of first-order lag term parameters that are optimized on the minimum (one-step) control horizon. As such, the NMPC strategy can be applied in more general cases concerning nonlinear and stochastic process/prediction models, e.g., for solving the AV safe speed control problem while interacting with pedestrians when approaching an unsignalized crosswalk. The PMPC strategy, particularly its filtered version (PMPCf), can approach the NMPC performance at a significantly reduced computational cost. However, due to its structural complexity, it may not be as equally attractive in more general nonlinear and/or stochastic dynamics cases as in the considered AV case study.

**Author Contributions:** Conceptualization, I.C., J.D., L.P., H.E.T., B.Š. and V.I.; methodology, I.C., L.P., J.D., H.E.T. and B.Š.; software, I.C. and L.P.; visualization, I.C. and L.P.; writing—original draft preparation, I.C., L.P. and B.Š.; writing—review and editing, J.D., I.C., L.P., B.Š., H.E.T. and V.I.; supervision, J.D., H.E.T. and V.I. All authors have read and agreed to the published version of the manuscript.

**Funding:** It is gratefully acknowledged that this work has been supported by Ford Motor Company.

**Data Availability Statement:** Not applicable.

**Acknowledgments:** It is gratefully acknowledged that the work of the third and fourth authors has been supported through the scientific–technological Croatian–Hungarian cooperation project "Design of Automated Driving Systems Based on Estimation of Wheel-road Contact Features for Handling Emergency Situations (AVEST)", and the work of the second author has been supported by the Croatian Science Foundation through the "Young researchers' career development project-training of new doctoral students".

**Conflicts of Interest:** The authors declare no conflict of interest.

## Abbreviations

| Abbreviation | Meaning |
|---|---|
| AV | Autonomous vehicle |
| MPC | Model predictive control |
| NMPC | Nonlinear model predictive control |
| PMPC | Parallel model predictive control |
| PMPCf | Parallel model predictive control filtered |
| MPC-MB | Move-blocking model predictive control |
| RMS | Root mean square |
| TLS | Traffic light signal |
| QP | Quadratic program |
| GLOSA | Green-light optimal speed advisory |
| LTV | Linear time varying |

## Appendix A. Linear Time-Varying Approach to Traffic-Light-Related Constraint

To transform the vehicle position and time-dependent traffic light constraint (4) into a time-dependent only constraint, the following algorithm is used.

The vehicle position constraint is not applied in the first simulation time step (controller initialization) since the predicted position vector is empty. In the next and subsequent sampling instances, the traffic light constraint (10) is based on a modified traffic light state sequence $\boldsymbol{S_{mod}}$, which is constructed from the actual traffic light state preview $S(h \mid k)$ and the predicted vehicle positions from the perspective of the previous time step, $s(h \mid k-1)$. This is similar to the linear time-varying MPC approach. The algorithm consists of the following steps:

1. **Step**-Initialize modified traffic light state prediction with actual traffic light state prediction:

$$S_{mod}(h|k) = S(h|k),\ \forall h \in \{0,\ 1, \ldots,\ N_p - 1\}$$

2. **Step**-Find the prediction time steps in which the vehicle crossed traffic light in the previous optimization step $k-1$ and store the result in vector $\boldsymbol{i_{cross}}$

$$i_{cross}(h-1|k) = \begin{cases} 0, & \text{for } s(h|k-1) < L_0 \\ 1, & \text{otherwise} \end{cases}$$

3. **Step**-Modify the traffic light state for the actual prediction horizon, using the following rules:

- If the vehicle did not cross the traffic light in the previous prediction, i.e., if $\boldsymbol{i}_{cross} = \mathbf{0}$, set the modified traffic light state to the actual one:

$$S_{mod}(h|k) = S(h|k),\ \forall h \in \{0,\ 1, \ldots,\ N_p - 1\}$$

- If the vehicle crossed the traffic light, i.e., if $\boldsymbol{i}_{cross} \neq \mathbf{0}$, find the first step at which the allowed crossing happened $h_c$, i.e., if $S(h_c \mid k) = 0$ and $i_{cross}(h_c \mid k) = 1$. Modify only the remaining part after the initial allowed crossing step $h_c$:

$$S_{mod}(h_c, \ldots, N_p - 1 \mid k) = 0$$

This modification considers all traffic light states after $h_c$ to be green, since the vehicle is going to cross during a green light, and all other traffic light states after the vehicle crosses are not relevant.

  - o The special case of $h_c = 0$ indicates that the vehicle just crossed the traffic light and that in the next time step the traffic light state should be considered green.
  - o In case the crossing is not allowed on the whole prediction horizon ($h_c$ is not found), the modified traffic light state prediction remains the same as the actual traffic light state prediction:

$$S_{mod}(h|k) = (h|k),\ \forall h \in \{0,\ 1, \ldots,\ N_p - 1\}$$

This means that the vehicle was not able to reach the crossing during the green traffic light phase, or the traffic light was in the red state for the whole duration of the prediction horizon. As soon as the next green traffic light phase is included on the horizon, the vehicle will be able to cross.

## Appendix B. Potential Infeasibility of the QP for Very Short Control Horizons $N_c << N_p$

For an extremely reduced control horizon in the linear MPC law ($N_c << N_p$), it is possible for the QP problem to become infeasible. A simple illustration is presented in Figure A1, where the traffic light is in the red state on the whole prediction horizon, and the vehicle is, therefore, not allowed to surpass the traffic light position (orange line). In this case, the initial vehicle velocity is quite high, and the vehicle needs to apply maximum deceleration for the whole $N_c$ duration, which is in this case the shortest feasible control horizon $N_c$*. For the remainder of the prediction horizon, the optimal acceleration control input is zero in order to stop before the traffic light. If the control horizon is further reduced, i.e., $N_c < N_c$*, the vehicle would not be able to stop in time, since the maximum deceleration must be applied during the whole $N_c$* time interval, and since $N_c < N_c$*, the vehicle would still be moving with some non-zero velocity at the end of $N_c$. Note that MPC could maintain small nonzero acceleration in the last step of the control horizon, and that input could hold for the remainder of the prediction horizon. However, that would cause the vehicle to surpass the traffic light position, meaning that in this case the vehicle is not able to safely stop using $N_c < N_c$*.

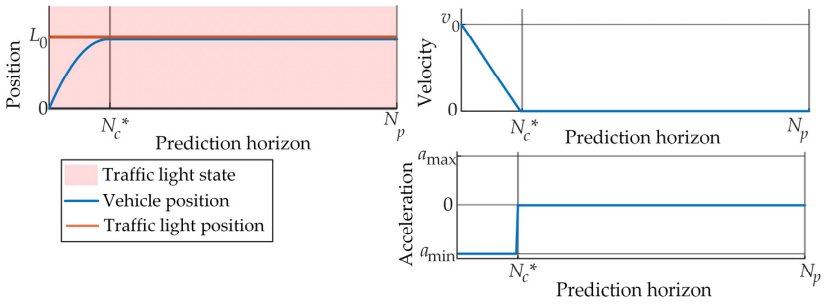


**Figure A1.** Illustration of the shortest control horizon case for feasible linear MPC with reduced control horizon ($N_c << N_p$).

## References


1. Texas A&M Transportation Institute 2019 Urban Air Mobility Report. Available online: https://mobility.tamu.edu/umr/report/#methodology (accessed on 15 December 2022).
2. Chada, S.K.; Purbai, A.; Gorges, D.; Ebert, A.; Teutsch, R. Ecological Adaptive Cruise Control for Urban Environments Using SPaT Information. In Proceedings of the 2020 IEEE Vehicle Power and Propulsion Conference (VPPC), Gijon, Spain, 18 November–16 December 2020; pp. 2–7.
3. Mintsis, E.; Vlahogianni, E.I.; Mitsakis, E. Dynamic Eco-Driving near Signalized Intersections: Systematic Review and Future Research Directions. *J. Transp. Eng. Part A Syst.* **2020**, *146*, 04020018. [CrossRef]
4. Lawitzky, A.; Wollherr, D.; Buss, M. Energy Optimal Control to Approach Traffic Lights. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 4382–4387.
5. Suzuki, H.; Marumo, Y. Safety Evaluation of Green Light Optimal Speed Advisory (GLOSA) System in Real-World Signalized Intersection. *J. Robot. Mechatron.* **2020**, *32*, 598–604. [CrossRef]
6. Held, M.; Flardh, O.; Martensson, J. Optimal Speed Control of a Heavy-Duty Vehicle in the Presence of Traffic Lights. In Proceedings of the 2018 IEEE Conference on Decision and Control (CDC), Miami, FL, USA, 17–19 December 2018; pp. 6119–6124.
7. Bouwman, K.R.; Pham, T.H.; Wilkins, S.; Hofman, T. Predictive Energy Management Strategy Including Traffic Flow Data for Hybrid Electric Vehicles. *IFAC-PapersOnLine* **2017**, *50*, 10046–10051. [CrossRef]
8. Bakibillah, A.S.M.; Kamal, M.A.S.; Tan, C.P.; Hayakawa, T.; Imura, J.I. Event-Driven Stochastic Eco-Driving Strategy at Signalized Intersections from Self-Driving Data. *IEEE Trans. Veh. Technol.* **2019**, *68*, 8557–8569. [CrossRef]
9. Asadi, B.; Vahidi, A. Predictive Cruise Control: Utilizing Upcoming Traffic Signal Information for Improving Fuel Economy and Reducing Trip Time. *IEEE Trans. Control Syst. Technol.* **2011**, *19*, 707–714. [CrossRef]
10. Xing, J.; Chu, L.; Guo, C. Optimization of Energy Consumption Based on Traffic Light Constraints and Dynamic Programming. *Electronics* **2021**, *10*, 2295. [CrossRef]
11. Kamalanathsharma, R.K.; Rakha, H.A. Multi-Stage Dynamic Programming Algorithm for Eco-Speed Control at Traffic Signalized Intersections. In Proceedings of the 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013), The Hague, The Netherlands, 6–9 October 2013; pp. 2094–2099.
12. Kamal, M.A.S.; Mukai, M.; Murata, J.; Kawabe, T. Model Predictive Control of Vehicles on Urban Roads for Improved Fuel Economy. *IEEE Trans. Control Syst. Technol.* **2013**, *21*, 831–841. [CrossRef]
13. Uebel, S.; Kutter, S.; Hipp, K.; Schrödel, F. A Computationally Efficient MPC for Green Light Optimal Speed Advisory of Highly Automated Vehicles. In Proceedings of the 5th International Conference on Vehicle Technology and Intelligent Transport Systems, Heraklion, Greece, 3–5 May 2019; pp. 444–451.
14. Huang, X.; Peng, H. Speed Trajectory Planning at Signalized Intersections Using Sequential Convex Optimization. In Proceedings of the 2017 American Control Conference (ACC), Seattle, WA, USA, 24–26 May 2017; pp. 2992–2997.
15. Pozzi, A.; Bae, S.; Choi, Y.; Borrelli, F.; Raimondo, D.M.; Moura, S. Ecological Velocity Planning through Signalized Intersections: A Deep Reinforcement Learning Approach. In Proceedings of the 2020 59th IEEE Conference on Decision and Control (CDC), Jeju, Republic of Korea, 14–18 December 2020; pp. 245–252.
16. Shao, Y.; Sun, Z. Eco-Approach with Traffic Prediction and Experimental Validation for Connected and Autonomous Vehicles. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 1562–1572. [CrossRef]
17. Ozatay, E.; Ozguner, U.; Filev, D.; Michelini, J. Analytical and Numerical Solutions for Energy Minimization of Road Vehicles with the Existence of Multiple Traffic Lights. In Proceedings of the 52nd IEEE Conference on Decision and Control, Firenze, Italy, 10–13 December 2013; pp. 7137–7142.
18. Meng, X.; Cassandras, C.G. Optimal Control of Autonomous Vehicles for Non-Stop Signalized Intersection Crossing. In Proceedings of the 2018 IEEE Conference on Decision and Control (CDC), Miami, FL, USA, 17–19 December 2018; pp. 6988–6993.
19. Dong, S.; Chen, H.; Yang, Z.; Liu, Q.; Wang, P. A Hierarchical Strategy for Velocity Optimization of Connected Vehicles with the Existence of Multiple Traffic Lights. In Proceedings of the 2019 Chinese Control Conference (CCC), Guangzhou, China, 27–30 July 2019; pp. 6680–6685.
20. De Nunzio, G.; De Wit, C.C.; Moulin, P.; Di Domenico, D. Eco-Driving in Urban Traffic Networks Using Traffic Signals Information. *Int. J. Robust Nonlinear Control* **2016**, *26*, 1307–1324. [CrossRef]
21. Mahmoud, Y.H.; Brown, N.E.; Motallebiaraghi, F.; Koelling, M.; Meyer, R.; Asher, Z.D.; Dontchev, A.; Kolmanovsky, I. Autonomous Eco-Driving with Traffic Light and Lead Vehicle Constraints: An Application of Best Constrained Interpolation. *IFAC-PapersOnLine* **2021**, *54*, 45–50. [CrossRef]
22. Mahler, G.; Vahidi, A. An Optimal Velocity-Planning Scheme for Vehicle Energy Efficiency through Probabilistic Prediction of Traffic-Signal Timing. *IEEE Trans. Intell. Transp. Syst.* **2014**, *15*, 2516–2523. [CrossRef]
23. Li, L.; Wen, D.; Yao, D. A Survey of Traffic Control with Vehicular Communications. *IEEE Trans. Intell. Transp. Syst.* **2014**, *15*, 425–432. [CrossRef]
24. Xu, B.; Ban, X.J.; Bian, Y.; Wang, J.; Li, K. V2I Based Cooperation between Traffic Signal and Approaching Automated Vehicles. In Proceedings of the 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA, 11–14 June 2017; pp. 1658–1664.
25. Van Middlesworth, M.; Dresner, K.; Stone, P. Replacing the Stop Sign: Unmanaged Intersection Control for Autonomous Vehicles. In Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS, Estoril, Portugal, 12–16 May 2008; Volume 3, pp. 1381–1384.

26. Jerez, J.L.; Constantinides, G.A.; Kerrigan, E.C.; Ling, K.V. Parallel MPC for Real-Time FPGA-Based Implementation. *IFAC Proc. Vol.* **2011**, *44*, 1338–1343. [CrossRef]
27. Jayaraman, S.K.; Robert, L.P.; Yang, X.J.; Pradhan, A.K.; Tilbury, D.M. Efficient Behavior-Aware Control of Automated Vehicles at Crosswalks Using Minimal Information Pedestrian Prediction Model. In Proceedings of the 2020 American Control Conference (ACC), Denver, CO, USA, 1–3 July 2020; pp. 4362–4368.
28. Škugor, B.; Deur, J.; Ivanovic, V.; Tseng, H.E. Stochastic Model Predictive Control of Autonomous Vehicles Approaching Unsignalized Crosswalks with Pedestrians. In Proceedings of the European Control Conference, Bucharest, Romania, 13–16 June 2023. *submitted for publication*.
29. Xiao, S.; Ge, X.; Han, Q.L.; Zhang, Y. Secure and Collision-Free Multi-Platoon Control of Automated Vehicles under Data Falsification Attacks. *Automatica* **2022**, *145*, 110531. [CrossRef]
30. Maciejowski, J.M. *Predictive Control: With Constraints*; Prentice Hall: Hoboken, NJ, USA, 2002; ISBN 0201398230.
31. Andersson, J.A.E.; Gillis, J.; Horn, G.; Rawlings, J.B.; Diehl, M. CasADi: A Software Framework for Nonlinear Optimization and Optimal Control. *Math. Program. Comput.* **2019**, *11*, 1–36. [CrossRef]
32. Ferreau, H.J.; Kirches, C.; Potschka, A.; Bock, H.G.; Diehl, M. QpOASES: A Parametric Active-Set Algorithm for Quadratic Programming. *Math. Program. Comput.* **2014**, *6*, 327–363. [CrossRef]
33. Cagienard, R.; Grieder, P.; Kerrigan, E.C.; Morari, M. Move Blocking Strategies in Receding Horizon Control. In Proceedings of the 2004 43rd IEEE Conference on Decision and Control (CDC), Nassau, Bahamas, 14–17 December 2004; pp. 2023–2028.
34. Grüne, L.; Pannek, J. *Nonlinear Model Predictive Control*, 1st ed.; Springer: London, UK, 2011; ISBN 978-0-85729-500-2.
35. Wächter, A.; Biegler, L.T. On the Implementation of an Interior-Point Filter Line-Search Algorithm for Large-Scale Nonlinear Programming. *Math. Program.* **2005**, *106*, 25–57. [CrossRef]