To appear in *Vehicle System Dynamics*
Vol. 00, No. 00, Month 20XX, 1–29

RESEARCH ARTICLE

A nonlinear model predictive control formulation for obstacle avoidance in high-speed autonomous ground vehicles in unstructured environments

Jiechao Liu[a], Paramsothy Jayakumar[b], Jeffrey L. Stein[c], and Tulga Ersal[d]*

[a] *Department of Mechanical Engineering, University of Michigan, G029 Walter E. Lay Automotive Laboratory, 1231 Beal Ave, Ann Arbor, MI, USA 48109; (734)763-7388; ljch@umich.edu*

[b] *U.S. Army RDECOM-TARDEC, 6501 E. 11 Mile Road, Warren, MI, USA 48397; paramsothy.jayakumar.civ@mail.mil*

[c] *Department of Mechanical Engineering, University of Michigan, 3439 George G. Brown Laboratory, 2350 Hayward, Ann Arbor, MI, USA 48109; (734)936-3336; stein@umich.edu*

[d] *Department of Mechanical Engineering, University of Michigan, G029 Walter E. Lay Automotive Laboratory, 1231 Beal Ave, Ann Arbor, MI, USA 48109; (734)763-7388; tersal@umich.edu*

(*February 2017*)

Acknowledgments:

The authors wish to acknowledge the financial support of the Automotive Research Center (ARC) in accordance with Cooperative Agreement W56HZV-14-2-0001 U.S. Army Tank Automotive Research, Development and Engineering Center (TARDEC) Warren, MI.

Word count:

- Abstract: 220 words
- Main text: about 7000 words

*Corresponding author. Email: tersal@umich.edu

This paper presents a nonlinear model predictive control (MPC) formulation for obstacle avoidance in high-speed, large-size autonomous ground vehicles (AGVs) with high center of gravity (CoG) that operate in unstructured environments, such as military vehicles. The term 'unstructured' in this context denotes that there are no lanes or traffic rules to follow. Existing MPC formulations for passenger vehicles in structured environments do not readily apply to this context. Thus, a new nonlinear MPC formulation is developed to navigate an AGV from its initial position to a target position at high speed safely. First, a new cost function formulation is used that aims to find the shortest path to the target position, since no reference trajectory exists in unstructured environments. Second, a region partitioning approach is used in conjunction with a multi-phase optimal control formulation to accommodate the complicated forms the obstacle-free region can assume due to the presence of multiple obstacles in the prediction horizon in an unstructured environment. Third, the no-wheel-lift-off condition, which is the major dynamical safety concern for high-speed, high-CoG AGVs, is ensured by limiting the steering angle within a range obtained offline using a 14 degrees-of-freedom vehicle dynamics model. Thus, a safe, high-speed navigation is enabled in an unstructured environment. Simulations of an AGV approaching multiple obstacles are provided to demonstrate the effectiveness of the algorithm.

Keywords: collision avoidance; vehicle dynamics; model predictive control; autonomous ground vehicles; vehicle safety

1. Introduction

The importance of autonomous ground vehicles (AGVs) has significantly increased over the past decades because they find many applications in both commercial and military fields. For large-size AGVs, i.e. vehicles that are at least the size of a passenger vehicle, dynamical limitations of the vehicles are critical factors to be considered in the development of the obstacle avoidance functions, especially at high speeds [1] to ensure dynamical safety. Therefore, obstacle avoidance algorithms must be able to operate AGVs at their dynamic limits when needed while still guaranteeing their dynamical safety.

To navigate vehicles among unexpected obstacles, many obstacle avoidance algorithms have been developed in the literature. In general, these algorithms can be classified into the following four categories: graph-search based methods [2, 3], virtual potential field and navigation function based methods [4, 5], meta-heuristic based methods [6], and mathematical optimization based methods [7, 8]. To address the abovementioned problem of interest, a mathematical optimization based approach is pursued in this work due to its generalizability and optimality. Furthermore, vehicle dynamics and various safety constraints can be explicitly considered in a rigorous and systematic way in this approach.

The mathematical optimization based approach can be further divided into two types. The first type is an open-loop strategy. An optimal path is generated from the initial point to the target point offline in advance. A feedback controller is then used online to follow the generated path. The second type is a closed-loop strategy. The optimization is performed online iteratively taking into account the state updates from the vehicle. One of the most well-established techniques in the closed-loop category is model predictive control (MPC) [9]. In this strategy, an optimal control problem (OCP) is formulated with a model of the system explicitly considered. Various input saturations and state constraints can also be simultaneously considered. The formulated OCP is then solved repeatedly over a finite receding horizon. The resulting control strategy is executed only until the next state updates are available, at which time a new control strategy is generated with updated state information. The successful applications of MPC to obstacle avoidance in AGVs have been demonstrated in prior research, including [10–15] among others.

Motivated by military applications, this paper studies the obstacle avoidance function of AGVs in unstructured environments. In the scope of this paper, the term 'unstructured' means that there are no lanes or traffic rules to follow. The AGV is commanded to move from its initial position to a specified target location fast and safely. Because mission performance is critical in military applications, the travel time is to be minimized. There exist obstacles between the initial and target positions. However, their sizes, shapes, and locations are not provided as an *a priori* knowledge to the navigation algorithm. A planar light detection and ranging (LIDAR) sensor is used to detect these obstacles. These considerations result in a context in which the existing MPC based obstacle avoidance algorithms cannot be readily applied because they mostly focus on on-road scenarios (e.g. [10]), which are structured environments due to the existence of lanes to be followed.

To clearly distinguish the scope of this work from the state-of-the-art MPC based obstacle avoidance algorithms, three major differences are summarized below. First of all, in a structured environment, a reference trajectory for the vehicle naturally exists, which is the centerline of the lane. When there is an unexpected obstacle on the road, it can be avoided by perturbing the reference trajectory. In contrast, in an unstructured environment, there is no such a reference trajectory. Thus, the obstacle avoidance algorithm needs to plan a path that not only avoids the obstacles but also leads the vehicle to a specified target position with only limited information.

Secondly, simple box constraints or constant bounds on position variables can be used to efficiently represent the obstacle-free regions (e.g. [11]) for on-road applications. These constraints can be constructed easily. However, in an unstructured environment, these types of constraints are not suitable because the obstacle-free regions become more complex especially when there are multiple obstacles of different sizes and shapes within the prediction horizon. Thus, to take the obstacle-free regions of complicated shapes into account in the MPC formulation, a systematic approach needs to be developed.

Finally, as mentioned before, the dynamical safety of the vehicle is a critical factor to be considered in the obstacle avoidance algorithm, especially when the vehicle travels at high speed. The dynamical safety can be interpreted differently in different applications. For passenger vehicles on slippery road or race cars, which are the focus of existing algorithms (e.g. [8]), the major dynamical safety concern is excessive sideslip. However, the focus of this paper is on vehicles with high center of gravity (CoG), such as military vehicles. For these types of vehicles, the major dynamical safety concern is wheel lift-off, which is a conservative requirement for preventing rollover. The existing obstacle avoidance algorithms do not explicitly include constraints to prevent wheel lift-off. Thus, a new MPC formulation is needed to prevent wheel lift-off of high-CoG AGVs while avoiding obstacles at high speeds.

In this paper, we extend the work presented in [16–18] and develop a novel nonlinear MPC based algorithm for obstacle avoidance of AGVs that can achieve an optimal and smooth operation of the vehicle through the obstacle field at high speed while ensuring vehicle safety; i.e., without wheel lift-off. The novelty of the paper compared to works such as [10–15] is three-fold: (1) This paper focuses on unstructured environments without a reference trajectory. A new cost function formulation is used that aims to find the shortest path to the target position in addition to approaching the target from a desired direction and minimizing the control effort. (2) To accommodate the complicated form of the obstacle-free region in the OCP formulation, the region is systematically partitioned, enabling a differentiable mathematical representation of the obstacle-free region and its inclusion in the OCP through a multi-phase approach. (3) The algorithm considers vehicles with relatively higher CoG and explicitly accounts for the vehicle

dynamical safety in terms of avoiding single wheel lift-off. This is achieved by limiting the steering angle within a range obtained offline using a 14 degrees of freedom (DoF) vehicle dynamics model. Simulations of an AGV in three different obstacle fields are given to demonstrate the effectiveness of the proposed algorithm.

Below is a list of assumptions to bound the scope of this work, which are discussed in detail later in this paper.

(1) All obstacles of interest are at least the height of where the LIDAR is mounted on the vehicle, which is in front of the vehicle.
(2) All obstacles are static.
(3) The vehicle longitudinal speed is maintained to be constant.
(4) The vehicle travels on a constant-friction flat surface.
(5) Vehicle parameters are constant.
(6) Vehicle state estimations are exact. No uncertainties are modeled.

Throughout this paper, the terms 'optimal trajectory', 'optimal states', and 'optimal control' are used with the understanding that they refer to the local optimal solution to which the OCP solver converges, and not to the global optimal solution. The local optimality is a result of the following two considerations. First, not all environmental information between the starting and target points is available to the algorithm at beginning; thus, a globally optimal solution cannot be expected. Second, the formulated OCP is nonconvex. Hence, the solution from the OCP solver may be a local optimal solution.

The rest of the paper is organized as follows. Section 2 briefly introduces the basic principle of MPC. Section 3 presents the formulation of the multi-phase OCP for obstacle avoidance, including the obstacle-free region partitioning approach, details of the vehicle dynamics model used, establishment of the maximum steering angle to meet the no-wheel-lift-off requirement, and solution techniques. Section 4 presents and discusses the simulation results. Section 5 provides a discussion of the algorithm including some important aspects that need further investigation. Conclusions are drawn in Section 6.

2. Basic Principle of MPC

The MPC is essentially a state-feedback controller with the control strategy calculated by solving an optimization problem. A model of the system is used explicitly to predict and optimize the system response over a finite receding horizon. At each time step, only the initial portion of generated control strategy is applied to the plant. At the next time step, a new OCP over a shifted receding horizon is solved with the initial states of the model being the measured state signals from the plant. A finite receding time horizon is used such that the controller can perform real-time optimization with hard constraints on plant variables [9].

Figure 1 illustrates the basic principle of the MPC concept. At the current time t_0, an open-loop constrained OCP is solved over a finite prediction horizon T_p with the model initialized by the state measurements. The solutions of the OCP include the optimal control sequence $\boldsymbol{\zeta}^*(t)$, $t \in [t_0, t_0 + T_p]$, and the estimated optimal state sequence $\boldsymbol{\xi}^*(t)$, $t \in [t_0, t_0 + T_p]$, which are generated based on both the current measurements and future predictions. The $\boldsymbol{\zeta}^*(t)$ is bounded by the input saturations and the $\boldsymbol{\xi}^*(t)$ satisfies the state constraints. Because the model used in the MPC is not an exact replica of the plant, the actual system state sequence $\boldsymbol{\xi}(t)$, $t \in [t_0, t_0 + T_e]$ is likely to deviate from the estimated sequence. Thus, at each time step, although $\boldsymbol{\zeta}^*(t)$ over the entire horizon

$t \in [t_0, t_0 + T_p]$ is generated, only the initial portion over the horizon $t \in [t_0, t_0 + T_e]$ is executed by the plant, where T_e is called the execution horizon. At the next time step $t_0 + T_e$, an OCP over a shifted prediction horizon $[t_0 + T_e, t_0 + T_e + T_p]$ is solved based on the new state measurements. As a result, the control strategy is updated every T_e seconds until the terminal requirements are satisfied. Because the state measurements are updated at each step of the MPC optimization, the closed-loop system is robust to some extent to uncertainties and noise in the feedback.

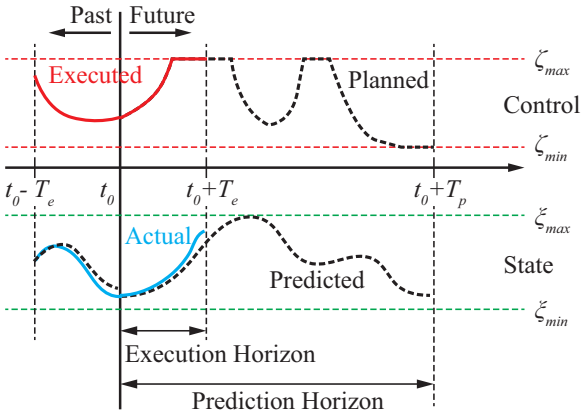


Figure 1. Basic principle of MPC.

The algorithm presented in this paper is within the nonlinear MPC domain. At each step, one or more multi-phase OCPs are solved as described later in Section 3.2. Throughout this paper, the term 'nonlinear MPC algorithm' and the term 'multi-phase optimal control algorithm' are used interchangeably.

3. Nonlinear MPC Algorithm for Obstacle Avoidance

The nonlinear MPC based obstacle avoidance algorithm is described in detail in this section. The algorithm and the plant forms a closed-loop system, the schematic of which is given in Figure 2. As shown in the figure, the nonlinear MPC algorithm has two parts, which are the LIDAR data processor and the control commands generator.

The nonlinear MPC algorithm requires three external inputs, which are the task information, the obstacle information, and the estimated state values. First of all, the task information is assumed to be given in this paper. Specifically, the task information includes the target location, the required final heading angle, and the reference vehicle speed. More discussion of the reference vehicle speed is presented in Section 5.

Secondly, the obstacle information is also required, which is obtained using a planar LIDAR sensor that is mounted in front of the vehicle. The output of the LIDAR is the distance to the closet obstacle boundary in each radial direction. The radial directions are defined by the angular range and angular resolution. In the paper, the sensor is assumed to have a 180° range and a resolution of ϵ. Thus, only the obstacles in front of the vehicle can be detected. The output of the LIDAR is the maximum detection range R_{LIDAR} when there are no obstacles within the detection range in that direction. In summary, the LIDAR provides information about the range and geometrical characteristics of the objects closest to the front of the vehicle. Figure 3 is an example output of the LIDAR sensor. There are three obstacles within the detection range and the vehicle heading is

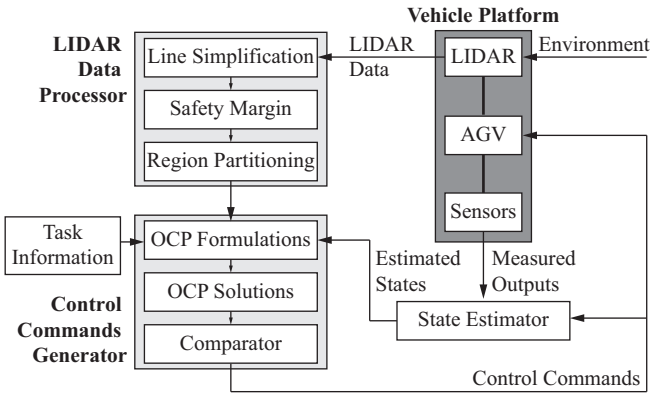
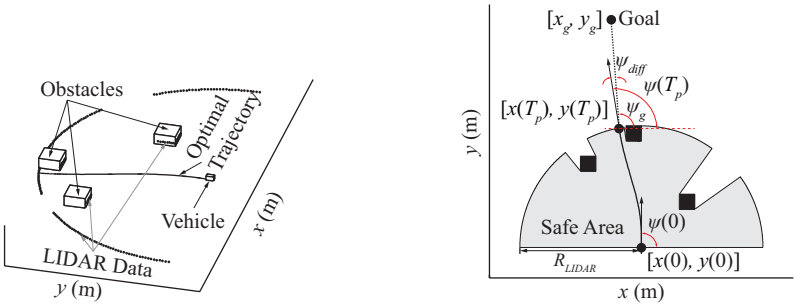


Figure 2. Schematic of the closed-loop system formed by the MPC-based obstacle avoidance algorithm and the AGV.



(a) A sample obstacle field with the LIDAR detection data plotted in the 3D space.

(b) The top view of the LIDAR data and the safe area. For the rest of this paper, it suffices to show only the top view because a planar LIDAR sensor is used.

Figure 3. LIDAR detection data [19].

parallel to the y-axis. As mentioned in Section 1, all obstacles are assumed static and high enough to be detected by the sensor.

Thirdly, at each step of the MPC, state measurements are used to initialize the vehicle model employed in the algorithm. Because a full-state feedback is required, a state estimator would be essential for a real application because not all states can be measured directly. However, since the AGV is simulated in this paper, a state estimator is not included because all signals are available from the plant model.

The LIDAR data processor uses the obstacle information and the state measurements to first generate a simplified obstacle shape using the raw LIDAR data. A safety margin is added afterwards and the resulting obstacle-free region is partitioned so that the constraints can be easily formulated. The control commands generator then formulates and solves OCPs to generate the optimal control commands using the task information, the state measurements, and the outputs of the LIDAR data processor. The details of these steps are described thoroughly in the following sub-sections.

3.1. *LIDAR Data Processor*

The LIDAR data processor processes a sequence of points defining the obstacle-free region into specifications of constraints that can be used in the OCP formulation. The obstacle-free region is considered as the safe region in which the vehicle can operate, hence the terms ‘obstacle-free region’ and ‘safe region’ are used interchangeably in this

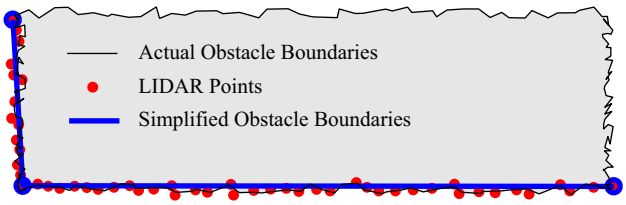


Figure 4. An illustration of the line simplification algorithm.

context. In this paper, a planar LIDAR is used and the set of points defining the safe region is simply the detected points by the LIDAR.

3.1.1. Line Simplification

The first step of the data processing is to reduce the number of points that define the obstacle boundaries for further processing, because the raw points obtained from the LIDAR can be noisy since the obstacle boundaries are not smooth and the detection results are not exact. In this work, the LIDAR noise is simulated by a uniform distribution of range [-0.1, 0.1] m. An algorithm is then required to identify the minimum number of lines for approximating the sequence of points considering the noise. The Ramer-Douglas-Peucker algorithm is an algorithm for reducing the number of points in a curve that is approximated by a series of points, which is widely used to perform simplification and denoising of range data acquired by a LIDAR [21] and has been adopted in this work, as well. As illustrated in Figure 4, the numerous detected points, which are generated using a simulated LIDAR with added noise, can be simplified into two line segments represented by three points, which provide a good approximation to the boundaries of the obstacle.

3.1.2. Safety Margin

A safety margin, l_{SM}, is added to the safe region to account for the size of the vehicle, detection noise, and differences between the predicted trajectory and the actual trajectory. Adding a safety margin allows for ignoring the vehicle size in the OCP formulation.

The safe area is a polygon in general and it is a simple polygon when the safe region boundary is obtained from a planar LIDAR sensor. Thus, algorithms for performing polygon offsetting (inflating/deflating) in computer graphics can be adopted to add the safety margin to the safe region. Specifically, the Vatti's clipping algorithm [22] implemented in the Clipper library [23] is used in this work.

The safety margin needs to be added to only specific segments of the safe region boundary. As shown in Figure 5, which corresponds to the example in Figure 3, the boundary of the safe region consists of three types of segments:

- LIDAR data segments, which specify the boundaries of the obstacles and are called 'obstacle boundaries'.
- Maximum LIDAR detection range segments, which are directions free from obstacles and are called 'openings'.
- Obstacle's laser shadow lines, which are along the rays from the LIDAR and are called 'hypothetical openings'. These lines are called 'hypothetical openings', because in its current position and orientation the vehicle cannot know whether they are actual openings or not because of the blocked view by the obstacles.

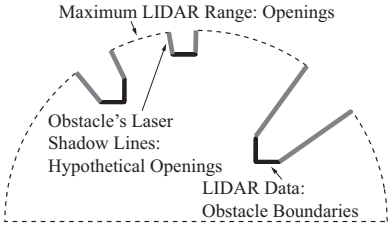


Figure 5. Three types of segments bounding the safe region.

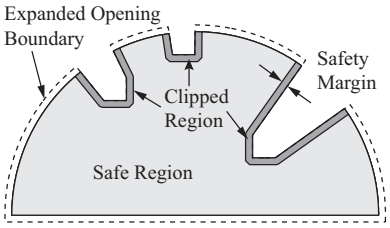


Figure 6. Safe region of Figure 5 with safety margin included.

The safety margin is to be applied to the obstacle boundaries and hypothetical openings only, but the Clipper library shrinks all the boundaries. Hence, the openings are first expanded by the same amount as the safety margin. The expanded region is then shrunk with the Clipper library, which effectively results in applying the safety margin only to the obstacle boundaries and hypothetical openings, and not to the openings. As an example, Figure 6 is the safe region with safety margin corresponding to Figure 5.

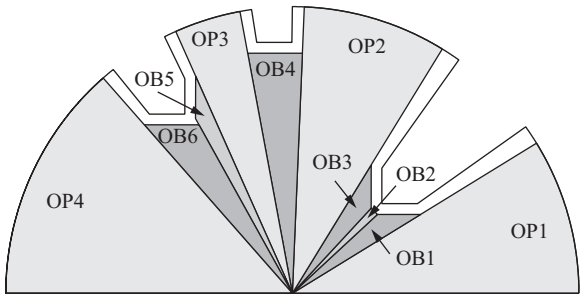
3.1.3. Region Partitioning

The safe region exemplified in Figure 6 is very difficult, if not impossible, to be represented mathematically using a single inequality in general. Even when the function exists, it is not differentiable at the connection points of the line segments from the first step. This would cause problems in the OCP solver that requires all functions to be twice continuously differentiable.

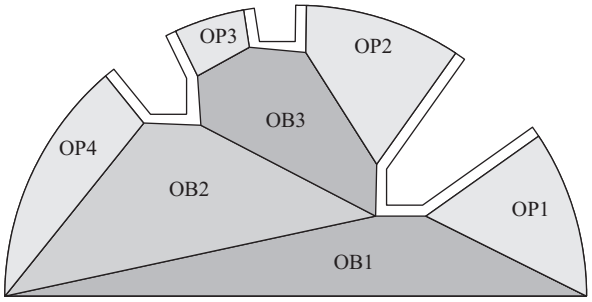
In order to represent the safe region in a mathematically suitable form for the OCP formulation, it is partitioned into several sub-regions, where each sub-region can be specified by a set of inequalities that are not piecewise functions and are differentiable. Two partitioning approaches are discussed below that can efficiently yield such sub-regions. Other partitioning approaches can be used, as well, as long as they provide efficient means for obtaining sub-regions that can be mathematically defined using non-piecewise and differentiable functions.

One approach is the 'polar partitioning'. The safe area is divided into sectors and triangles, where sectors are regions including an opening and triangles are regions including an obstacle boundary. Figure 7a is the partitioning of the safe region shown in Figure 6 using this approach, where 'OP' and 'OB' denote the regions that terminate with an opening and obstacle boundary, respectively. As an example, Region 'OB4' is a triangle, which can be specified using three linear inequalities, whereas Region 'OP3' is a sector, which is bounded by two lines. The third boundary of Region 'OP3' is an arc; however, because of the limits on prediction horizon, this arc constraint will never be active and thus is ignored. Polar partitioning is easy to implement, because the original LIDAR data is in polar coordinates.

The second approach is the 'optimal convex partitioning' or simply 'convex



(a) Polar partitioning.



(b) Convex partitioning.

Figure 7. Example partitioning of the safe region in Figure 6 using two approaches [19]

partitioning'. The interior of the safe area is decomposed into a minimum number of convex regions without introducing additional points inside the polygon. Specifically, the dynamic programming algorithm in [24] is incorporated into this work, which is very efficient in decomposing simple polygons. In this approach, 'OP' denotes the regions that terminate with an opening, whereas 'OB' denotes all other regions. Figure 7b is an exemplified partitioning for the safe region in Figure 6 using the convex partitioning approach. Similar to the polar partitioning approach, all the regions can be specified using a set of linear inequalities after partitioning. This approach is capable of partitioning a safe region in a more general form. Thus, the algorithm is not limited to be used with a planar LIDAR sensor. As long as a safe region can be deduced from the sensor data, the algorithm is applicable.

In either approach, a sub-region can be defined by

$$\begin{bmatrix} \vdots & \vdots \\ a_j & b_j \\ \vdots & \vdots \end{bmatrix}^{(i)} \begin{bmatrix} x^{(i)} \\ y^{(i)} \end{bmatrix} \leq \begin{bmatrix} \vdots \\ c_j \\ \vdots \end{bmatrix}^{(i)} , j = 1, \ldots, L \tag{1}$$

where i is the sub-region index and L is the total number of line segments bounding that sub-region; a_j, b_j and c_j are coefficients calculated based on the two end points of the corresponding line segments; (x, y) is a position in Cartesian coordinates.

Equation (1) can be compacted in the following form:

$$\boldsymbol{A}^{(i)}_{L\times 1} x^{(i)}(t) + \boldsymbol{B}^{(i)}_{L\times 1} y^{(i)}(t) \leq \boldsymbol{C}^{(i)}_{L\times 1}, t \in [T^{i-1}, T^i] \tag{2}$$

where $\boldsymbol{A}^{(i)}_{L\times 1}$ is a vector with the jth entry being a_j. The definitions of $\boldsymbol{B}^{(i)}_{L\times 1}$ and $\boldsymbol{C}^{(i)}_{L\times 1}$ are similar.

To avoid collision with the obstacles and move towards the target, the trajectory should stay within the safe region and the last part of the predicted trajectory should lie within the sub-region of type 'OP'. The following list presents the procedures of forming the constraints in the OCP formulation to meet the above requirement. The safe region partitioning shown in Figure 7b is used as an example to elaborate.

(1) Identify the first region to traverse ('SR' (starting region)). Region 'OB1' is the 'SR' in this example, because the current position of the vehicle is located on one of its boundaries.
(2) Identify all regions with a feasible opening ('TRs' (terminal regions)). A feasible opening is an arc segment, which can have an intersection with a feasible trajectory over a slightly longer period than the prediction horizon. They can be identified using the two extreme trajectories, which are shown in Figure 8. The two extreme trajectories are obtained by simulating a 2 DoF vehicle model using steering controls that are at the limits of handling at each step at the measured initial states. In the example, the arc segments in 'OP2' and 'OP3' are feasible openings and hence they are 'TRs', whereas 'OP1' and 'OP4' are not 'TRs', because the vehicle cannot make a sharp enough turn to move into those partitions safely.
(3) Find the sequence of regions from the 'SR' to a 'TR' for all 'TRs'. This is a shortest path problem, where many algorithms can be used, including Dijkstra and A*. Dijkstra's algorithm [25] is chosen here, because it is one of the simplest solutions. For example, with 'OP3' being the 'TR', the region sequence is identified as 'OB1 → OB2 → OB3 → OP3'. Figure 8 highlights this region sequence. With 'OP2' being the 'TR', the region sequence is identified as 'OB1 → OB2 → OB3 → OP2'.

For a region sequence from the 'polar partitioning' as exemplified in Figure 9a, a different region partition as shown in Figure 9b can be obtained easily. This alternative partition approach is preferred when one of the boundaries separating two regions is almost along the vehicle heading direction.

The specifications of the regions with this partitioning approach are given by

$$\begin{aligned} & R_{\min}^{(i)} \leq \sqrt{[x^{(i)}(t) - x(0)]^2 + [y^{(i)}(t) - y(0)]^2} \leq R_{\max}^{(i)} \\ & \Phi_{\min}^{(i)} \leq \operatorname{atan2}(y^{(i)}(t) - y(0), x^{(i)}(t) - x(0)) \leq \Phi_{\max}^{(i)} \\ & t \in [T^{i-1}, T^i] \end{aligned} \tag{3}$$

where $R_{\min}^{(i)}$, $R_{\max}^{(i)}$, $\Phi_{\min}^{(i)}$, and $\Phi_{\max}^{(i)}$ are bounds calculated from the coordinates of end points specifying a region.

In the simulations for this paper, the convex partition approach is used primarily, and the polar partition approach is used secondarily as a failsafe approach in case the optimal solution to the problem formulated using the convex partition cannot be obtained.

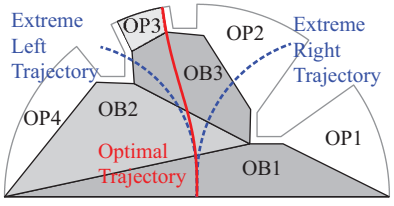
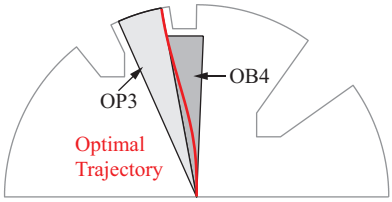
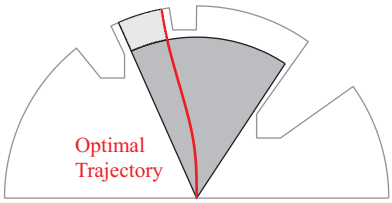


Figure 8. An example of extreme trajectories and region sequence.



(a) Basic polar partitioning.



(b) Modified polar partitioning.

Figure 9. Regions from the polar partitioning approach and its variance.

Once the partitioning is performed, two potential OCP formulation approaches can be pursued, a mixed integer nonlinear programming (NLP) approach [26–28] or a multi-phase NLP approach [29–31]. The latter approach is utilized in this work due to its efficiency and is described in detail next.

3.2. *Control Command Generator*

At each step of the MPC, there may exist multiple feasible openings. It may be possible to use some heuristics to choose the best feasible opening based on the vehicle state and target position, however, a rigorous approach is used in this work. In particular, for each feasible opening, a sequence of regions the vehicle needs to move through is obtained using Dijkstra's algorithm. Thus, a multi-phase OCP is formulated for each feasible opening. These formulated problems are solved in parallel. The final solution is obtained by comparing the solutions of these problems and picking the one that gives the minimum cost. In summary, one or more OCPs are formulated and solved at each step. The formulation in general form is given in Equation (4) - Equation (10) below.

By minimizing the cost function specified in Equation (4), subject to constraints defined by Equation (5) - Equation (10) for all phases, the optimal state trajectories $\boldsymbol{\xi}^{*(i)}(t), t \in [T^{i-1}, T^i]$, the optimal control trajectories $\boldsymbol{\zeta}^{*(i)}(t), t \in [T^{i-1}, T^i]$, and the times T^{i-1}, T^i, $i = 1, \ldots, N$ for transitioning from one region to the next are obtained, where N is the total number of phases, which is the number of regions from 'SR' to the 'TR'. The details of the formulation are given in the following subsection.

$$\underset{\boldsymbol{\xi},\boldsymbol{\zeta},T^1,\ldots,T^N}{\text{minimize}} \quad J = \mathcal{T}\left[\boldsymbol{\xi}^{(N)}\left(T^N\right),\boldsymbol{\zeta}^{(N)}\left(T^N\right),T^N\right] + \sum_{i=1}^{N}\left\{\int_{T^{i-1}}^{T^i}\mathcal{I}\left[\boldsymbol{\xi}^{(i)}\left(t\right),\boldsymbol{\zeta}^{(i)}\left(t\right)\right]\mathrm{d}t\right\} \tag{4}$$

$$\underset{\forall i=1,\ldots,N}{\text{subject to}} \quad \dot{\boldsymbol{\xi}}^{(i)}(t) = \boldsymbol{\mathcal{V}}\left[\boldsymbol{\xi}^{(i)}(t),\boldsymbol{\zeta}^{(i)}(t)\right] \tag{5}$$

$$\boldsymbol{\xi}^{(i)}(T^{i-1}) = \boldsymbol{\xi}^{(i-1)}(T^{i-1}), \boldsymbol{\xi}^{(0)}(T^0) = \boldsymbol{\xi_0} \tag{6}$$

$$\boldsymbol{\mathcal{S}}^{(i)}\left[x^{(i)}(t),y^{(i)}(t)\right] \leq 0 \tag{7}$$

$$\delta_{f,\min}(U_0) \leq \delta_f^{(i)}(t) \leq \delta_{f,\max}(U_0) \tag{8}$$

$$\varsigma_{f,\min} \leq \varsigma_f^{(i)}(t) \leq \varsigma_{f,\max} \tag{9}$$

$$\begin{aligned} &t \in [T^{i-1},T^i], T^{i-1} < T^i \\ &T^0 = 0, T^N = T_p, T_{p,\min} \leq T_p \leq T_{p,\max} \end{aligned} \tag{10}$$

3.2.1. Cost Function

The cost function defines the soft requirement, i.e., in what sense the trajectory is optimal. Since a reference trajectory does not exist in this work, the cost function aims to find the shortest path. If the task is only to pass a target location without a specific direction requirement, the trajectory is optimal when the end point of the predicted trajectory is close to the target, and the final heading angle is pointing to the target, because a shorter distance-to-go is preferred to minimize travel time. The cost function for this case is defined as

$$J = s_T/s_0 + w_\psi \psi_{\text{diff}}^2 + w_d d \tag{11}$$

where

$$s_0 = \sqrt{[x_g - x(0)]^2 + [y_g - y(0)]^2} \tag{12}$$

$$s_T = \sqrt{[x_g - x(T_p)]^2 + [y_g - y(T_p)]^2} \tag{13}$$

$$\psi_{\text{frg}} = \text{atan2}\left(y_g - y(T_p), x_g - x(T_p)\right) \tag{14}$$

$$\psi_{\text{diff}} = \text{atan2}\left(\sin(\psi(T_p) - \psi_{\text{frg}}), \cos(\psi(T_p) - \psi_{\text{frg}})\right) \tag{15}$$

$$d = \int_0^{T_p}\left[\varsigma_f^2(t) + w_\delta \delta_f^2(t)\right]\,\mathrm{d}t \tag{16}$$

Specifically, the cost function formulation includes three terms that are linearly combined using relative weights, w_ψ and w_d.

The first term is a ratio between s_T and s_0, where s_0 is the distance between the initial position $[x(0), y(0)]$ and the target position $[x_g, y_g]$ as defined in Equation (12), and s_T is the distance between the end point of the predicted trajectory $[x(T_p), y(T_p)]$

and the target as defined in Equation (13). Visual representations of all variables are shown in Figure 3b. The second term is the difference between the final heading angle $\psi(T_P)$ and the angle of the target relative to the end point of the predicted trajectory ψ_{frg} as defined in Equation (15). The weighted sum of the first two terms is used to lead the vehicle to the specified target position from its current position. This weighted sum captures the distance-to-go with the assumption that there exist no obstacles beyond the sensed horizon. The third term is a regularization term minimizing the control effort d as defined in Equation (16), where ς_f is the steering rate, which is the control command to be optimized, δ_f is the front wheel steering angle, and w_δ is a weight.

If a particular direction of passing the target location in global coordinates is also required, the following cost function is used.

$$J = s_T/s_0 + w_\psi \psi_{\text{diff}}^2 + w_s s + w_d d \tag{17}$$

where

$$s = \int_0^{T_p} [l_a x(t) + l_b y(t) + l_c]^2 \, \mathrm{d}t \tag{18}$$

$$\begin{aligned} l_a &= \sin(\psi_g), l_b = -\cos(\psi_g) \\ l_c &= -\sin(\psi_g) x_g + \cos(\psi_g) y_g \end{aligned} \tag{19}$$

The cost function specified by Equation (17) has one more term than Equation (11). This term represents the integral of the distance to the line given by $l_a x + l_b y + l_c = 0$ over the prediction horizon. This line passes through the target position $[x_g, y_g]$ along the desired direction ψ_g.

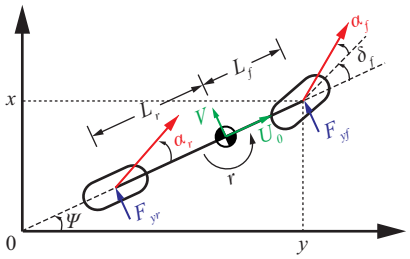
When the target position is within the sensor's detection range, the term s_T/s_0 and $w_\psi \psi_{\text{diff}}^2$ are removed from the cost functions given in Equation (11) and Equation (17). Instead the following constraints are added to the OCP formulation.

$$\begin{aligned} x_g - \sigma \leq x(T_p) \leq x_g + \sigma \\ y_g - \sigma \leq y(T_p) \leq y_g + \sigma \end{aligned} \tag{20}$$

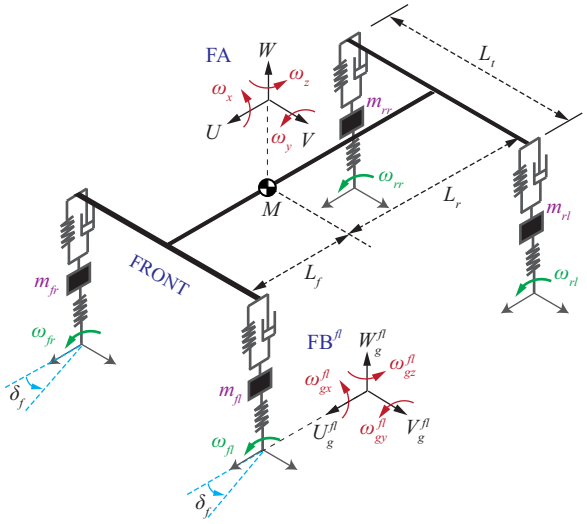
where σ is a small margin. If the vehicle is within this margin from the target position, then the target is considered to be reached.

3.2.2. Vehicle Models

Two different vehicle dynamics models are used in this work: a 14 DoF model to represent the plant and to generate offline the dynamic-safety-related look-up table used in the MPC, and a 2 DoF model used in the MPC to predict vehicle trajectories. The schematics of the two representations are shown in Figure 10. The 14 DoF model consists of a single sprung mass connected to four unsprung masses. The suspensions between the sprung mass and unsprung masses are modeled as spring-damper systems. In the 2 DoF model, the left and right tires on each axle are lumped together. The equations for the 14 DoF model are omitted here for space limitations, but can be found in the literature [32].



(a) Two DoF vehicle model.



(b) Fourteen DoF vehicle model.

Figure 10. Schematics of the vehicle models. The 14 DoF model is used to represent the actual vehicle in simulations, whereas the 2 DoF model is used for the trajectory predictions in the MPC.

The 2 DoF model is described by the following ODEs.

$$\dot{V} = (F_{y,f} + F_{y,r})/M - U_0 r \tag{21}$$

$$\dot{r} = (F_{y,f} L_f - F_{y,r} L_r)/I_{zz} \tag{22}$$

$$\dot{\psi} = r \tag{23}$$

$$\dot{x} = U_0 \cos\psi - (V + L_f r)\sin\psi \tag{24}$$

$$\dot{y} = U_0 \sin\psi + (V + L_f r)\cos\psi \tag{25}$$

where $F_{y,f}$ and $F_{y,r}$ are tire lateral forces generated at the front axle and the rear axle, respectively. U_0 and V are the longitudinal speed and lateral speed in the vehicle's coordinate frame, respectively. r is the yaw rate, ψ is the yaw angle, (x, y) is the vehicle's front center location in global coordinates, M is the vehicle mass, I_{zz} is the moment of inertia, L_f is the distance between the front axle and the vehicle's CoG location, and L_r is the distance between the rear axle and the vehicle's CoG location.

By using the Pacejka Magic Formula ($\mathcal{P}$) tire model with pure slip, the lateral tire forces are represented as

$$F_{y,f} = \mathcal{P}_y(F_{z,f}, \alpha_f) \tag{26}$$

$$F_{y,r} = \mathcal{P}_y(F_{z,r}, \alpha_r) \tag{27}$$

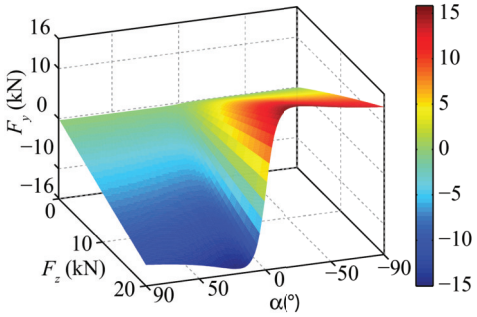


Figure 11. Lateral tire force as a function of vertical load and slip angle described by the Pacejka magic formula tire model.

The exact form of the Pacejka tire model can be found in [33]. Figure 11 shows the relationship between the tire lateral forces and the slip angle at different vertical loads described by the Pacejka tire model. In the 14 DoF vehicle model, the Pacejka Magic Formula tire model with combined slip is used. Note that the Pacejka tire model is an empirical expression and is not limited to only pneumatic tires on pavement, but can also represent the force slip characteristics of pneumatic tires on soft cohesive soils [34].

As concluded in [17], it is important to account for the longitudinal load transfer in the 2 DoF vehicle model when the vehicle travels at high speed. Thus, the following relationships are used with the 2 DoF model to calculate the vertical loads on the front and rear axles taking into account the longitudinal load transfer effects.

$$F_{z,f} = (MgL_r + MVrh_{\mathrm{CG}})/(L_f + L_r) \tag{28}$$

$$F_{z,r} = (MgL_f - MVrh_{\mathrm{CG}})/(L_f + L_r) \tag{29}$$

where h_{CG} is the height of the vehicle CoG location above the ground.

In addition, the slip angles of front and rear tires are obtained from

$$\alpha_f = \tan^{-1}\left[(V + L_f r)/(U_0) - \delta_f\right] \tag{30}$$

$$\alpha_r = \tan^{-1}\left[(V - L_r r)/(U_0)\right] \tag{31}$$

The steering rate ς_f is used as the control command to be optimized and the steering angle δ_f is set as an additional state variable of the system. The reason for choosing the steering rate as the control input instead of the steering angle is to obtain a smooth steering angle sequence and impose a limit on the steering rate.

By setting the state vector as $\boldsymbol{\xi} = [\,x\ y\ \psi\ V\ r\ \delta_f\,]$ and the control vector as $\boldsymbol{\zeta} = \varsigma_f$, the state-space equation for the 2 DoF nonlinear vehicle model can be written as

$$\dot{\boldsymbol{\xi}} = \boldsymbol{f}(\boldsymbol{\xi}) + \boldsymbol{B}\boldsymbol{\zeta} \tag{32}$$

where

$$\boldsymbol{f}(\boldsymbol{\xi}) = \begin{bmatrix} U_0\cos\psi - (V + L_f r)\sin\psi \\ U_0\sin\psi + (V + L_f r)\cos\psi \\ r \\ (F_{y,f} + F_{y,r})/M - U_0 r \\ (F_{y,f}L_f - F_{y,r}L_r)/I_{zz} \\ 0 \end{bmatrix}$$

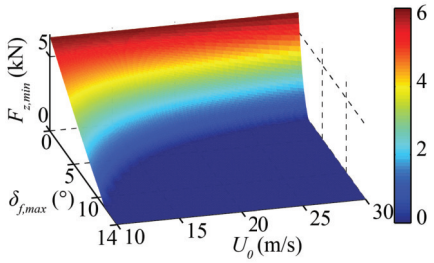


Figure 12. Minimum tire vertical load at different combinations of vehicle speed and maximum steering angle.

$$\boldsymbol{B}^T = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The 2 DoF vehicle model with longitudinal load transfer and nonlinear tire model is a very good approximation to the 14 DoF vehicle model within the prediction horizon length considered for this work [17].

3.2.3. Obstacle Avoidance

Obstacle avoidance is enforced through the constraint that the vehicle trajectory must lie within the safe region. For each of the phases in the multi-phase OCP, the vectors $\boldsymbol{A}^{(i)}_{L\times 1}$, $\boldsymbol{B}^{(i)}_{L\times 1}$, and $\boldsymbol{C}^{(i)}_{L\times 1}$ or the bounds $R^{(i)}_{\min}$, $R^{(i)}_{\max}$, $\Phi^{(i)}_{\min}$, and $\Phi^{(i)}_{\max}$ can be calculated using the values stored in the structure variable SafeRegion. The specific form of Equation (7) is either given in Equation (2) or Equation (3) depending on the scenario.

3.2.4. Dynamical Safety

In this study, ensuring the vehicle's dynamical safety is translated to avoiding single wheel lift-off. This is a conservative criterion used to prevent rollover [35]. For the constant-speed scenario assumed in this paper, the dynamical safety requirement is imposed through an upper bound on the steering angle magnitude as expressed by the following inequality constraint.

$$|\delta_f(t)| \leq \delta_{f,\max}(U_0) \tag{33}$$

where the maximum steering angle $\delta_{f,\max}$ is a function of only the vehicle speed when the vehicle is assumed to move on a flat surface.

For all combinations of longitudinal speed ranging from 10 m/s to 30 m/s and maximum steering angle ranging from 0° to 14°, the corresponding minimum tire vertical loads are obtained using the 14 DoF vehicle model. To capture the highly transient conditions in extreme maneuvers, the minimum tire vertical load for a given steering angle δ_f is found by initially steering the vehicle with $-\delta_f$ and then changing to δ_f with the maximum steering rate. The relationship between the maximum steering angle, longitudinal speed, and minimum tire vertical load is shown in Figure 12. If a minimum vertical load threshold is set, the relationship between the maximum steering angle and longitudinal speed can be extracted. Figure 13 shows the relationship when $F_{z,\min}$ is set as 500 N.

3.2.5. Solution Techniques

A two-step procedure is used to solve the nonlinear multi-phase OCPs presented in Section 3.2. The first step is to convert the continuous OCP into an NLP problem. To

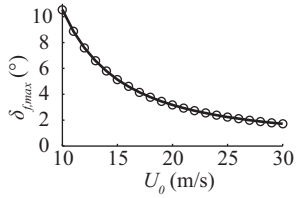


Figure 13. Maximum steering angle as a function of vehicle longitudinal speed when the minimum vertical load threshold is 500 N.

this end, a direct method called *hp*-pseudospectral method [29–31] is leveraged. The second step is to solve the resulting NLP problem, for which the interior point method [36] is used. Both of the methods are briefly reviewed below for completeness.

The *hp*-pseudospectral method has been demonstrated to be computationally efficient in accurately approximating the solution to a general continuous-time OCP [31]. It uses a variable number of intervals and a variable-degree polynomial within each interval to approximate the continuous state and control variables with discrete sequences. The Legendre-Gauss-Radau (LGR) quadrature points are used as the collocation points in this work. The differential-algebraic constraints of the OCP are enforced at these locations. Thus, a continuous-time OCP is transcribed into an NLP problem.

After transforming from the time interval $t \in [0, T_p]$ to the time interval $\tau \in [-1, 1]$ via the following variable transformation

$$t = (T_p/2)(\tau + 1) \tag{34}$$

the state $\boldsymbol{\xi}$ is approximated by a polynomial of degree at most n as follows

$$\boldsymbol{\xi}(\tau) \approx \sum_{p=1}^{n} \boldsymbol{\Xi}_p L_p(\tau), \quad L_p(\tau) = \prod_{q=1, q\neq p}^{n} \frac{\tau - \tau_q}{\tau_p - \tau_q} \tag{35}$$

where $\tau_p (p = 1, \ldots, n)$ are the LGR collocation points, $L_p(\tau)(p = 1, \ldots, n)$ are the bases of the Lagrange polynomials, and $\boldsymbol{\Xi}_p$ is the state approximation at τ_p.

For example, after discretization, Equation (5) that represents the dynamic model of the vehicle using a set of first-order ODEs can be converted into the following sets of equality constraints to ensure the dynamical feasibility of the results

$$\Xi_p^{(i)} - \Xi_0^{(i)} - \frac{T^{(i-1)} - T^{(i)}}{2} \sum_{q=1}^{n} A_{pq}^{(i)} \mathcal{V}\left[\Xi_q^{(i)}, Z_q^{(i)}\right] = 0, p = 1, \ldots, n \tag{36}$$

where A is called the integration matrix, which is defined by the selected LGR collocation points and the corresponding weights [31].

The primal-dual interior-point algorithm with a filter line search method implemented in IPOPT [36] is used to solve the NLP problem. In brief, the key concept of the algorithm is using a barrier function and a barrier parameter, which help to decompose the NLP problem with both equality and inequality constraints into a series of NLP problems with only equality constraints. These equality-constrained problems can then be solved iteratively using the Newton-Raphson method to determine the search direction and the backtracking line search method to obtain the step size.

The interior point method converts the general NLP problem given in Equation (37)

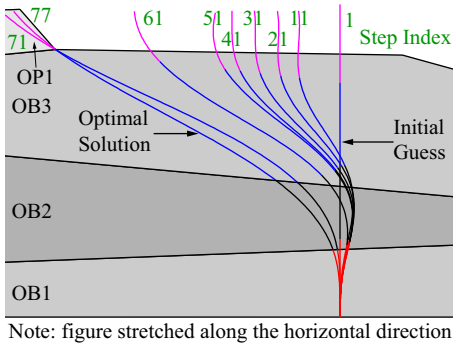


Figure 14. The trajectory iterations from the initial guess to the optimal solution for the example in Figure 8.

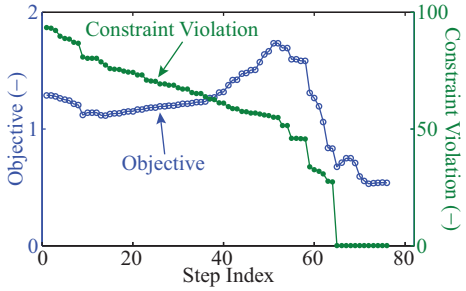


Figure 15. The objective and constraint violation at all steps during the iterations in Figure 14.

to a series of NLPs with only equality constraints given in Equation (38).

$$\underset{Z\in\mathbb{N}^n}{\text{minimize}} \quad f(Z) \tag{37}$$

$$\text{subject to} \quad \boldsymbol{C}(Z) = 0$$

$$Z \geq 0$$

$$\underset{Z\in\mathbb{N}^n}{\text{minimize}} \quad f(Z) + \mu_k B(Z) \tag{38}$$

$$\text{subject to} \quad \boldsymbol{C}(Z) = 0$$

where μ is a small positive scalar called 'barrier parameter'. As μ converges to zero, the solution to Equation (38) should converge to a solution to Equation (37). $B(\boldsymbol{.})$ is a barrier function.

As an example, Figure 14 shows the trajectory iterations in solving the problem shown in Figure 8. This is a four-phase problem. The initial guess is a straight line assuming equal length at each phase, which is not a feasible solution. Nevertheless, after 77 iterations, the solution converges to the optimal solution. Figure 15 shows the corresponding objective value and maximum constraint violation at all steps.

4. Simulation Results

This section presents numerical simulations of the developed nonlinear MPC based obstacle avoidance algorithm with a 14 DoF vehicle model as the plant. The algorithm is implemented in MATLAB® for proof-of-concept. Table 1 gives all the parameters used by the 2 DoF vehicle model.

Table 1. Vehicle Parameters

Parameter	Value
M (kg)	2252
m (kg)	110
I_{zz} (kg-m^2)	4110
L_f (m)	1.58
L_r (m)	1.72
L_t (m)	1.82
$h_{\mathrm{CG}}(m)$	1.00

Three scenarios are considered in this section. In the first scenario, the vehicle is required to move from its initial position to a target position with the final heading angle required to be the same as the initial heading angle. Two obstacles are between the initial and target locations. Vehicle speeds ranging from 10 m/s to 30 m/s are considered.

In the second scenario, the vehicle has to traverse a dense obstacle field to reach the target position. There are 50 obstacles and each of them is 10 m $\times$ 10 m in size. The vehicle longitudinal speed is maintained at 20 m/s and there is no constraint on the final heading angle.

In the third scenario, the vehicle performs a NATO double lane change maneuver at 15 m/s using the obstacle avoidance algorithm.

Table 2 summarizes the weighting parameters in the cost function used for all scenarios. If there is no constraint on the final heading angle, the weight w_s is not used. Thus, in the first and third scenario, all four weighting parameters are used. However, in the second scenario, w_s is omitted.

Table 2. Weighting Parameters

Parameter	Value
w_ϕ	1
w_d	10
w_δ	0.1
w_s	10^{-4}

4.1. *Scenario 1: Various Speeds*

Table 3 summarizes the parameters of the nonlinear MPC algorithm, including safety margin, LIDAR detection range, length of prediction horizon, length of execution horizon, and maximum steering angle. The cost function given in Equation (17) is used, because the angle of passing the target position is specified in this scenario. In the settings, the following relationship is used, which ensures that all the predicted trajectories lie within the LIDAR detection range.

$$T_{p,\max} = R_{\mathrm{LIDAR}}/U_0 \tag{39}$$

The execution horizon specifies the update rate of the algorithm. In this work, it is defined as one fifteenth of the prediction horizon. In Table 3, the values of execution horizon in terms of time are different at different vehicle speeds. However, for a given sensor detection range, the execution horizon in terms of distance is fixed.

Table 3. Algorithm Parameters

U_0 (m/s)	10	15	20	25	30	30
l_{SM} (m)	3	3	3	3	3	3
R_{LIDAR} (m)	100	100	100	100	100	140
$T_{p,\max}$ (s)	10.0	6.7	5.0	4.0	3.3	4.7
T_e (s)	0.67	0.44	0.33	0.27	0.22	0.31
$\varsigma_{f,\max}$ ($^\circ/s$)	10	10	10	10	10	10
$\delta_{f,\max}$ ($^\circ$)	10.5	5.14	3.18	2.24	1.72	1.72

The first set of simulations uses a LIDAR with a detection range of 100 m. The results of the simulations are presented in Figure 16.

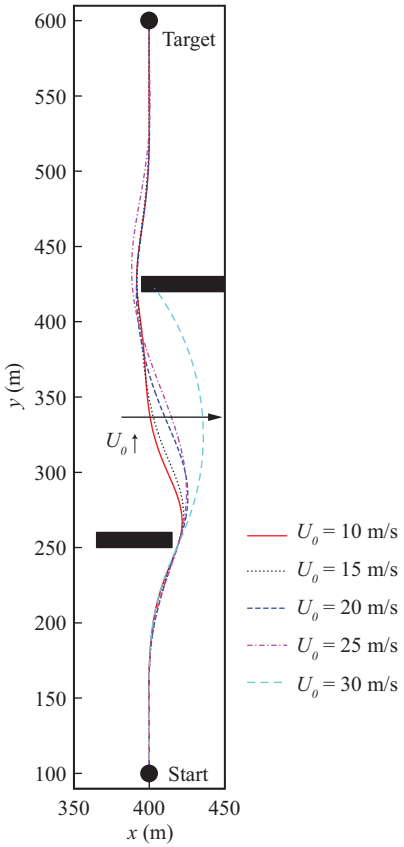
These results show that the developed algorithm can successfully navigate the vehicle through the specified obstacle field at 10 m/s, 15 m/s, 20 m/s, and 25 m/s. At these speeds, the vehicle avoids all obstacles, passes the target from the desired direction, and is dynamically safe as shown in Figure 16c. However, the vehicle hits the second obstacle when the longitudinal speed is maintained at 30 m/s. This is because at this speed, the vehicle is not capable of making a turn at a smaller radius safely. A threshold of 500 N is set on the minimum tire vertical load and the corresponding maximum steering angle is set as a hard constraint in the OCP formulation. This constraint is active at most of the time during the maneuver as shown in Figure 16b.

The navigation at 30 m/s fails because the LIDAR detection range is not long enough to accommodate that speed and hence the prediction horizon is too short to prepare the vehicle to avoid the obstacles sufficiently early. This is an example to demonstrate the limitation of the presented algorithm. One potential solution is to use a sensor with a longer range, if the speed is to be maintained constant. Figure 17 shows the results of simulations with different LIDAR detection ranges. When a longer detection range of 140 m and a longer prediction horizon is used, the vehicle travels through the field safely. Another potential solution is to extend the formulation to also consider the longitudinal speed, but this is beyond the scope of this paper.

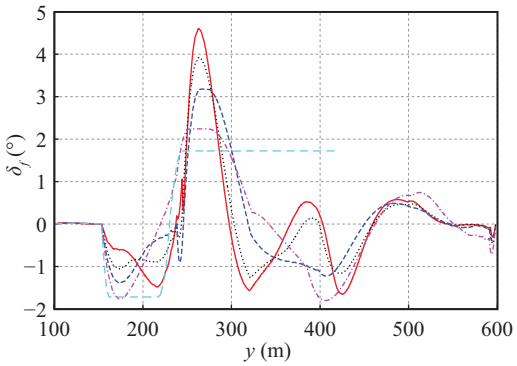
4.2. *Scenario 2: Dense Obstacle Field*

This simulation is to test the capability of the algorithm within a dense obstacle field. In this simulation, the vehicle speed is maintained at 20 m/s and there is no constraint on the final heading angle. Hence, Equation (11) is used as the cost function and the algorithm parameters are the same as the ones corresponding to 20 m/s in Table 3. Figure 18 shows the simulation results. The vehicle clears the obstacle field and reaches the target successfully using the algorithm.

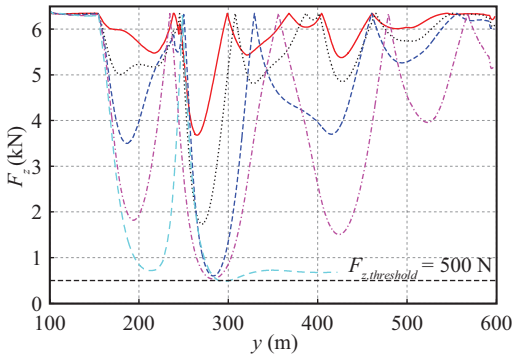
In this scenario, at most of the steps, there are multiple feasible openings as exemplified by Figure 19. For each of the feasible openings, an OCP is formulated and solved. After all of them are solved, their objective function values are compared and the one with the smallest value is considered the best solution. In this example, the objective values of the calculated trajectories from right to left are 0.76, 0.74, 0.92, respectively. The smallest one is 0.74 and the control commands corresponding to the trajectory in the middle is sent to the plant.



(a) Trajectory.



(b) Steering angle.



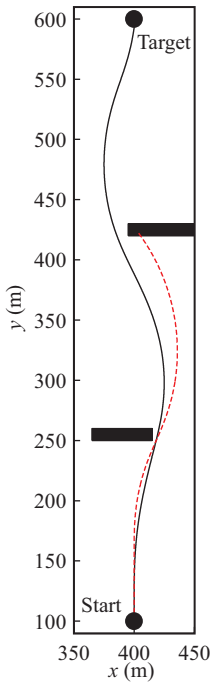
(c) Minimum tire vertical load.

Figure 16. Results of simulations with various longitudinal speeds.

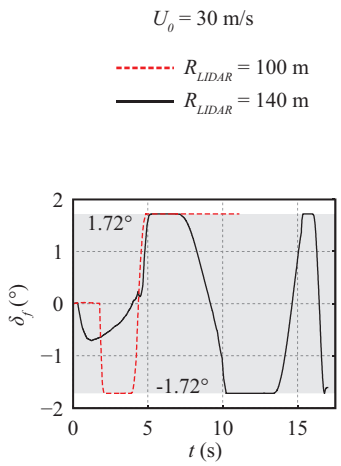
4.3. *Scenario 3: Double Lane Change*

Although this paper is concerned with unstructured environments, the proposed algorithm can also work for structured environments. The last simulation is to demonstrate the capability of the algorithm in a structured environment. The vehicle performs a double lane change maneuver at 15 m/s using the nonlinear MPC algorithm. Table 4 summarizes the parameters used.

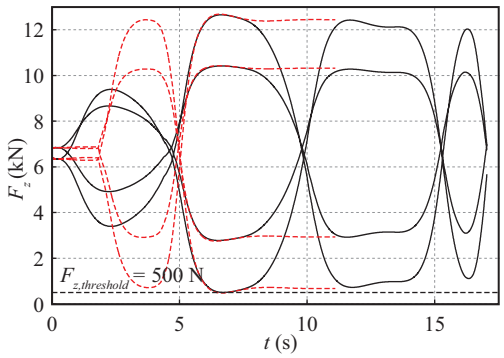
Figure 20 shows the generated trajectory and the corresponding steering angle. Figure 20a shows the trajectory of the CoG of the plant and the corresponding trajectories of the four corners of the vehicle. It can be seen that all the trajectories are within the white space, which means that the vehicle is free from collision. Figure 20b is



(a) Vehicle trajectory.



(b) Steering angle. Shaded area represents the allowable values.



(c) Tire vertical loads.

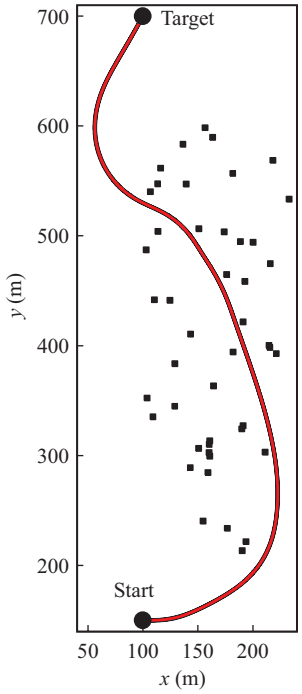
Figure 17. Results of simulations with different LIDAR detection ranges at 30 m/s.

the corresponding steering sequence.

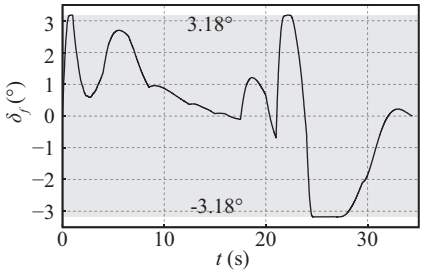
In this scenario, in most of the steps, there are no 'openings' as defined in Figure 5. However, there are 'hypothetical openings' as defined in Section 3.1.2. The 'TRs' are then defined as all regions with a feasible hypothetical opening. Figure 21 shows the use of a hypothetical opening.

5. Discussion

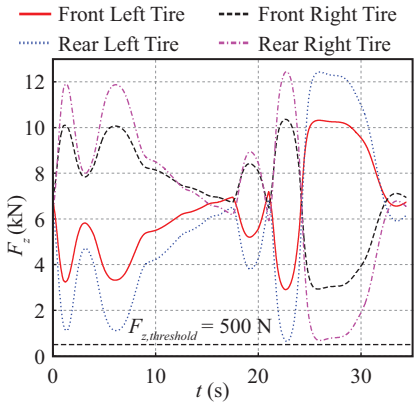
This paper assumes that the obstacles are static. This assumption is suitable for the context of this work considering the fact that in an unstructured environment AGVs typically encounter static obstacles more often. Nevertheless, extension to moving obstacles would certainly broaden the utility of the algorithm and is a recommended direction for further research.



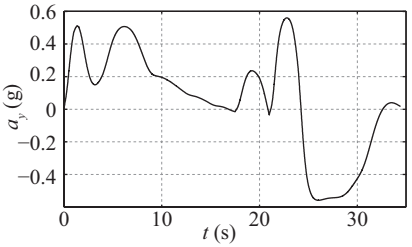
(a) Vehicle trajectory.



(b) Steering angle. Shaded area represents the allowable values.



(c) Tire vertical loads.



(d) Lateral acceleration.

Figure 18. Simulation results of navigation within a dense obstacle field.

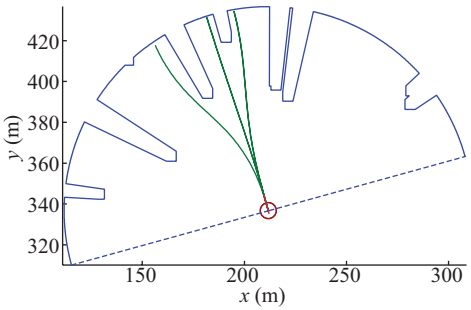
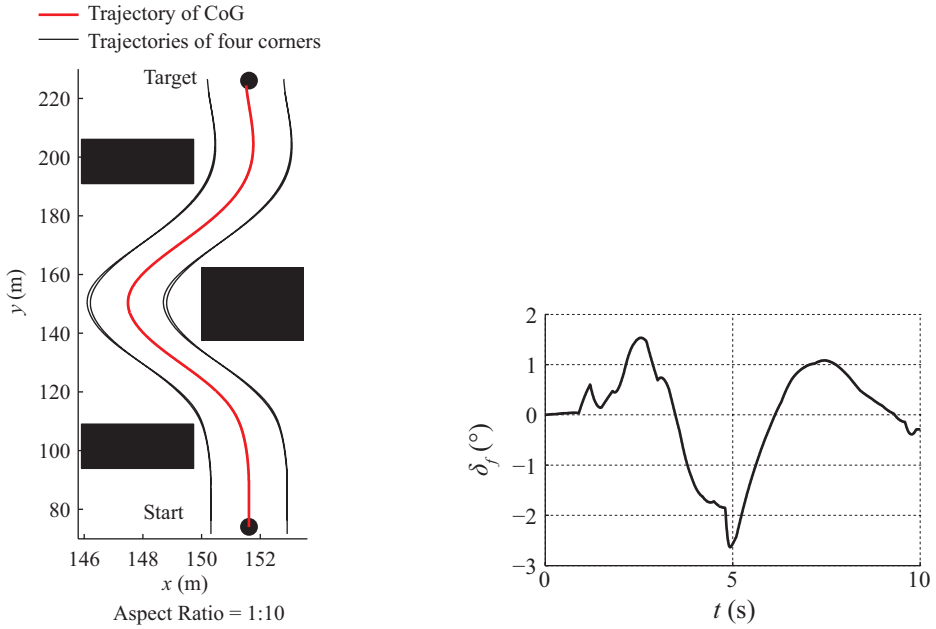


Figure 19. Example instant with multiple feasible openings in scenario 2.

Table 4. Algorithm Parameters for Double Lane Change Test

Parameter	Value
l_{SM} (m)	1.6
R_{LIDAR} (m)	50
$T_{p,\max}$ (s)	3.0
T_e (s)	0.3
$\varsigma_{f,\max}$ ($^\circ/s$)	10
$\delta_{f,\max}$ ($^\circ$)	5.14



(a) Trajectory. (b) Steering angle.

Figure 20. Simulation results of double lane change maneuver.

This paper also assumes that the vehicle speed is provided to the algorithm as an input. The obstacle avoidance algorithm as presented in this paper is not capable of determining the maximum speed that can be used to safely navigate through an obstacle field, because the on-board sensors provide information about the environment within only the close proximity of the vehicle. However, a conservative lower bound on the prediction horizon or a conservative lower bound on the sensor detection range can be imposed to ensure that the obstacle avoidance maneuver is performed early enough. These limits could be obtained from the trajectory for making a 90° turn when the initial steering angle is at the minimum bound, which is considered as the most extreme maneuver. For example, the trajectories from speeds ranging from 10 m/s to 30 m/s are shown in Figure 22. The time of completing this maneuver is considered as the minimum prediction time, which is summarized in Figure 23a. According to Equation (39), the minimum detection range is given in Figure 23b. As shown in the figure, the speed should be limited below 17 m/s when the LIDAR detection range is 100 m if this conservative bound is used. The third option is to let the MPC algorithm control both the steering and vehicle speed. It is worth noting that the OCP formulation presented in this paper can be extended to include the vehicle speed as another control variable in addition to steering [19, 20].

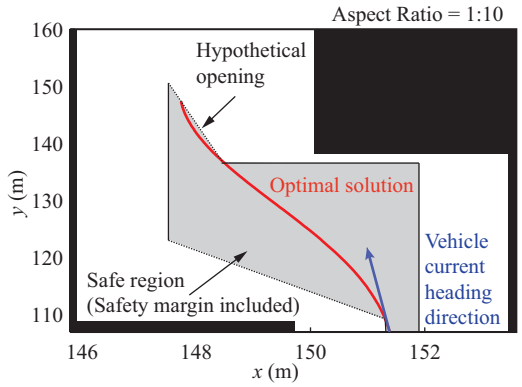


Figure 21. Example usage of a hypothetical opening in scenario 3.

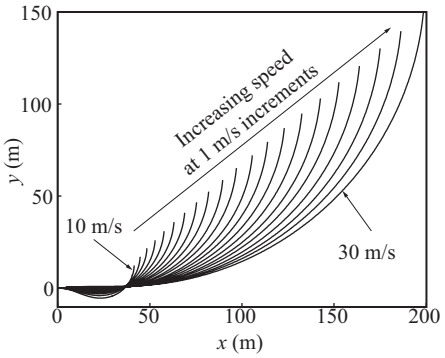
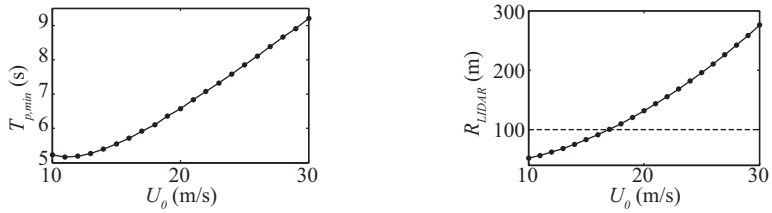


Figure 22. The trajectories of vehicle making a 90° turn at various speeds.



(a) Minimum prediction time. (b) Minimum detection range

Figure 23. Limits on parameters to ensure early maneuver.

In real systems, there may be several locations where time delays are introduced. Such delays can be lumped into two groups: the sensing delay and the control delay. These delays can have significant impact on the performance of the algorithm and hence need to be captured in the control loop. In a separate work, we have proposed methods to explicitly take delays into account to increase the robustness of the algorithm [18]. Specifically, the sensing delay is compensated by taking into account the differences of vehicle position and heading between the time when the LIDAR sensor is obtained and the time when the sensor data is used. The control delay is compensated by simulating a delayed control sequence, predicting the vehicle states after the delay with a vehicle model, and using the predicted states as initial states for planning.

To generate a differentiable mathematical representation of the safe region from LIDAR data, the region is first partitioned as described in Section 3.1.3. If the safe region could be used directly in the OCP formulation, the resulting optimal solution is expected to be the same as the optimal solution of the multi-phase OCP. Although a formal proof

is not available, the fact that the same solution is obtained with different partitioning approaches could be a support for this argument. For example, the trajectories shown in Figure 8, Figure 9a, and Figure 9b are the same, although they are obtained using three different partitioning approaches.

There are four weighting parameters in the cost function. Note that all scenarios presented in this paper are run with the same set of weighting parameters, indicating that the algorithm can cover different scenarios with the same set of weights. It is also interesting to point out how different choices of weights can affect the performance in a given scenario. As an example, the scenario 1 with vehicle speed at 20 m/s is ran with five sets of weighting parameters listed in Table 5.

Table 5. Weighting Parameters

	w_ϕ	w_d	w_δ	w_s
Set 1	1	10	0.1	10^{-4}
Set 2	10	10	0.1	10^{-4}
Set 3	1	100	0.1	10^{-4}
Set 4	1	10	1.0	10^{-4}
Set 5	1	10	0.1	10^{-3}

As shown in Figure 24a and Figure 24b, all vehicle trajectories are collision free and all steering angle trajectories are within the maximum allowed bounds, which means that the vehicle is dynamically safe for all cases. Thus, the algorithm can be utilized with a wide range of weighting parameters.

Because all terms in the cost function have intuitive meanings, it is easy to adjust the value of the weighting parameters for different purposes. If the weight w_d is increased, a smaller control effort is expected, which will result in turns with larger turning radius. If the weight w_s is increased, the trajectory will tend to follow the line passing through the target location from the specified heading direction. Thus, shaper turns are expected. Simulation results in Figure 24 agree with these expectations.

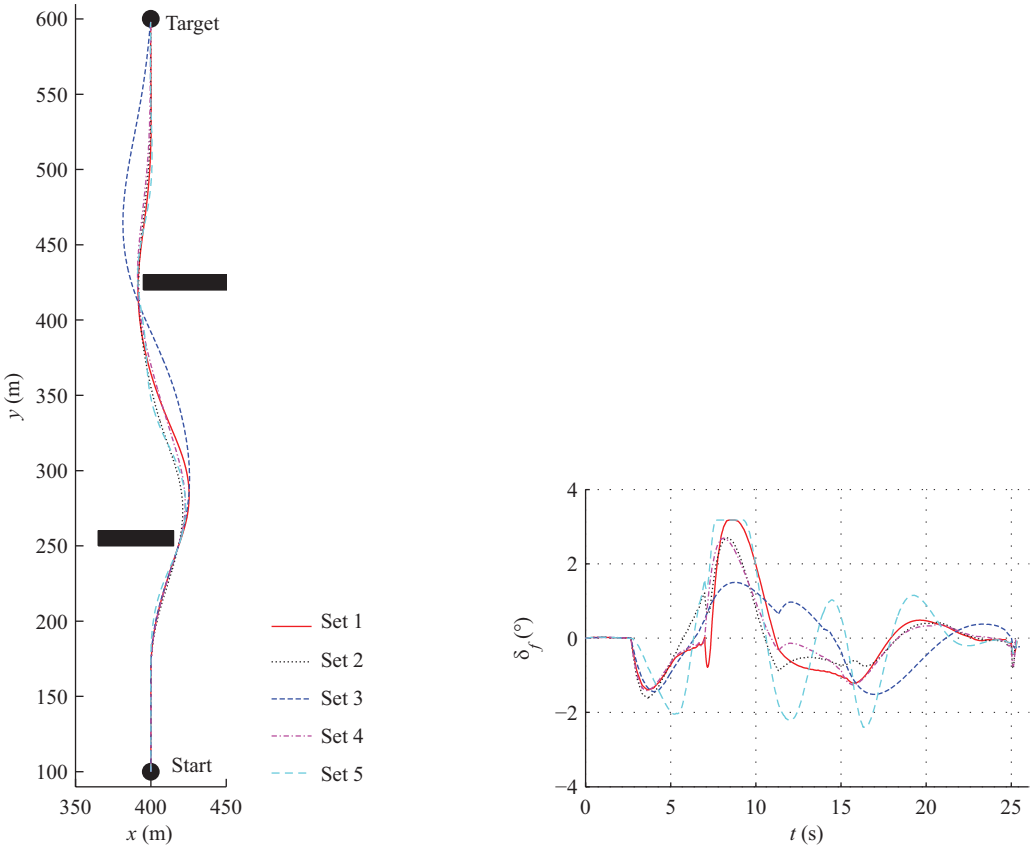
As stated in Section 4, the algorithm is currently implemented in MATLAB® for proof-of-concept. Thus, it does not yet run in real-time for all cases. Specifically, it is estimated to be about 10 times slower than real-time in the worst case with a 3.5 GHz Intel® Xeon® processor. However, when fully implemented in a compiled language with an optimized code, the algorithm is expected to run in real time.

When solving the OCP, only the instantaneous LIDAR data are used in the present formulation; prior data are not stored. Even though this approach is sufficient to navigate the vehicle safely in the scenarios presented in this paper, tracking the LIDAR data in time could improve performance.

Finally, in the simulations the LIDAR range was varied up to 140 m. This is a reasonable range to consider given the range of the state-of-the-art sensors that can extend to 200 m [37].

6. Conclusion and Future Work

This paper presents a novel nonlinear MPC algorithm for obstacle avoidance in high-speed, large-size AGVs with high center of gravity that operate in unstructured environments, such as military vehicles. A systematic approach is presented to partition the obstacle-free region obtained from an on-board LIDAR sensor into multiple sub-regions that have differentiable mathematical representations. A multi-phase OCP



(a) Trajectory. (b) Steering angle.

Figure 24. Simulation results to study the weighting parameters in the cost function.

formulation is then adopted to accommodate the different position constraints introduced by each sub-region. The no-wheel-lift-off requirement, which is the main dynamical safety concern for high-speed vehicles with a high CoG, is satisfied by limiting the steering angle within a range that is a function of vehicle speed. This range is obtained by simulating a 14 DoF vehicle dynamics model offline. The cost function is formulated in a way to allow the vehicle reach the target position faster and, if needed, from a desired angle, and with minimal control effort. Simulation results show that the method can yield a satisfactory performance in a variety of scenarios. The conclusion is that the method can enable a safe navigation of high-speed AGVs through an unstructured obstacle field, where safety is understood in terms of avoiding not only the obstacles, but also wheel lift-off. Future research directions include enabling combined lateral and longitudinal control, handling parametric uncertainties, moving obstacles, and sloping ground, and pursuing a real-time implementation of the algorithm.

7. Acknowledgment

The authors wish to acknowledge the financial support of the Automotive Research Center (ARC) in accordance with Cooperative Agreement W56HZV-14-2-0001 U.S. Army Tank Automotive Research, Development and Engineering Center (TARDEC) Warren, MI.

References

[1] Beal CE, Gerdes JC. Model predictive control for vehicle stabilization at the limits of handling. IEEE Trans. Control Syst. Technol. 2013;21:1258-1269.

[2] LaValle SM, Planning algorithms. New York (NY): Cambridge University Press; 2006.

[3] Luders BD, Karaman S, How JP. Robust sampling-based motion planning with asymptotic optimality guarantees. AIAA Guidance, Navigation, and Control Conference; 2013.

[4] Khatib O. Real-time obstacle avoidance for manipulators and mobile robots. Int. J. Rob. Res. 1986;5:90-98

[5] Shimoda S, Kuroda Y, Iagnemma K. High-speed navigation of unmanned ground vehicles on uneven terrain using potential fields. Robotica. 2007;25:409-424.

[6] Hussein A, Mostafa H, Badrel-Din M, et. al. Meta-heuristic optimization approach to mobile robot path planning. International Conference on Engineering and Technology; 2012.

[7] Ogren P, Leonard NE. A convergent dynamic window approach to obstacle avoidance. IEEE Trans. Rob. 2005;21:188-195.

[8] Gao Y, Lin T, Borrelli F, et al. Predictive control of autonomous ground vehicles with obstacle avoidance on slippery roads. Dynamic Systems and Control Conference; 2010.

[9] Allgwer F, Zheng A. Nonlinear model predictive control. Birkhuser Verlag, Basel, Switzerland; 2000.

[10] Park JM, Kim DW, Yoon YS, et al. Obstacle avoidance of autonomous vehicles based on model predictive control. J. of Automobile Eng. 2009;223:1499-1516.

[11] Bevan GP, Gollee H, O'Reilly J. Trajectory generation for road vehicle obstacle avoidance using convex optimization. J. of Automobile Eng. 2010;224:455-473.

[12] Tahirovic A, Magnani G. General framework for mobile robot navigation using passivity-based MPC. IEEE Trans. Autom. Control. 2011;56:184-190.

[13] Gray A, Gao Y, Lin T, et al. Predictive control for agile semi-autonomous ground vehicles using motion primitives. American Control Conference; 2012.

[14] Frasch JV, Gray A, Zanon M, et al. An auto-generated nonlinear MPC algorithm for real-time obstacle avoidance of ground vehicles. European Control Conference; 2013.

[15] Jeon JH, Cowlagi RV, Peters SC, et al. Optimal motion planning with the half-car dynamical model for autonomous high-speed driving. American Control Conference; 2013.

[16] Liu J, Jayakumar P, Overholt JL, et al. The role of model fidelity in model predictive control based hazard avoidance in unmanned ground vehicles using LIDAR sensors. Dynamic Systems and Control Conference; 2013.

[17] Liu J, Jayakumar P, Stein JL, et al. A study on model fidelity for model predictive control-based obstacle avoidance in high-speed autonomous ground vehicles. Vehicle Syst. Dyn. 2016;54(11):1629-1650.

[18] Liu J, Jayakumar P, Stein JL, et al. A multi-stage optimization formulation for MPC-based obstacle avoidance in autonomous vehicles using a LIDAR sensor. Dynamic Systems and Control Conference; 2014.

[19] Liu J, Jayakumar P, Stein JL, et al. An MPC algorithm with combined speed and steering control for obstacle avoidance in autonomous ground vehicles. Dynamic Systems and Control Conference; 2015.

[20] Liu J, Jayakumar P, Stein JL, et al. Combined speed and steering control in high speed autonomous ground vehicles for obstacle avoidance using model predictive control. IEEE Trans. Veh. Technol. 2017;99:8746-8763.

[21] Douglas DH, Peucker TK. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. Cartographica. 1973;10:112-122.

[22] Vatti BR. A generic solution to polygon clipping. Comm. of ACM. 1992;35:56-63.

[23] Johnson A. 2015, Clipper - an open source freeware library for clipping and offsetting lines and polygons. Accessed 20 November 2015. http://www.angusj.com/delphi/clipper.php

[24] Keil M, Snoeyink J. On the time bound for convex decomposition of simple polygons. Int. J. Comput. Geom. Ap. 2002;12:181-192.

[25] Dijkstra EW. A note on two problems in connexion with graphs. Num. Math. 1959;1:269-271.

[26] Kuwata Y, How JP. Receding Horizon Implementation of MILP for Vehicle Guidance. American Control Conference; 2005.

[27] Achterberg T. SCIP: solving constraint integer programs. Math. Program. Comput. 2009;1:1-41.

[28] Deits R, Tedrake R. Efficient mixed-integer planning for UAVs in cluttered environments. IEEE International Conference on Robotics and Automation; 2015.

[29] Rao AV. A survey of numerical methods for optimal control. AAS/AIAA Astrodynamics Specialist

Conference; 2010.
[30] Darby CL, Hager WW, Rao AV. An hp-adaptive pseudospectral method for solving optimal control problems. Optim. Contr. Appl. Met. 2011;32:476-502.
[31] Darby CL. hp-pseudospectral method for solving continuous-time nonlinear optimal control problems [dissertation]. University of Florida; 2011.
[32] Shim T, Ghike C. Understanding the limitations of different vehicle models for roll dynamics studies. Veh. Syst. Dyn. 2007;45:191-216.
[33] Pacejka HB, Tire and vehicle dynamics. Oxford (UK): Butterworth-Heinemann; 2005.
[34] Maclaurin B. Using a modified version of the Magic Formula to describe the traction/slip relationships of tyres in soft cohesive soils. J Terramechanics. 2014;52:1-7.
[35] Chen BC, Peng H. Rollover warning for articulated heavy vehicles based on a time-to-rollover metric. J. Dyn. Syst. Meas. Contr. 2005;127:406-414.
[36] Waechter A, Biegler LT. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math. Program. 2006;106:25-57.
[37] AutonomousStuff. 2015, ibeo LUX 8L (Prototype) Technical facts. Accessed 20 November 2015. http://www.autonomoustuff.com/uploads/9/6/0/5/9605198/ibeo_lux_as.pdf.