





Article

Limited Visibility Aware Motion Planning for Autonomous Valet Parking Using Reachable Set Estimation

Seongjin Lee ¹, Wontek Lim ¹, MyoungHo Sunwoo ¹ and Kichun Jo ^{2,*}

¹ Department of Automotive Engineering, Hanyang University, 222 Wangsimni-ro, Seongdong-gu, Seoul 04763, Korea; seongjin90@gmail.com (S.L.); lwt1849@gmail.com (W.L.); msunwoo@hanyang.ac.kr (M.S.)

² Department of Smart Vehicle Engineering, Konkuk University, Seoul 05029, Korea

* Correspondence: kichun.jo@gmail.com; Tel.: +82-2-2049-6265

Abstract: Autonomous driving helps drivers avoid paying attention to keeping to a lane or keeping a distance from the vehicle ahead. However, the autonomous driving is limited by the need to park upon the completion of driving. In this sense, automated valet parking (AVP) system is one of the promising technologies for enabling drivers to free themselves from the burden of parking. Nevertheless, the driver must continuously monitor the automated system in the current automation level. The main reason for monitoring the automation system is due to the limited sensor range and occlusions. For safety reasons, the current field of view must be taken into account, as well as to ensure comfort and to avoid unexpected and harsh reactions. Unfortunately, due to parked vehicles and structures, the field of view in a parking lot is not sufficient for considering new obstacles coming out of occluded areas. To solve this problem, we propose a method that estimates the risks for unobservable obstacles by considering worst-case assumptions. With this method, we can ensure to not act overcautiously while moving safe. As a result, the proposed method can be a proactive approach to consider the limited visibility encountered in a parking lot. In the proposed method, occlusion can be efficiently reflected in the planning process. The potential of the proposed method is evaluated in a variety of simulations.

Keywords: automated valet parking; planning under uncertainty; replanning; utility theory



Citation: Lee, S.; Lim, W.; Sunwoo, M.; Jo, K. Limited Visibility Aware Motion Planning for Autonomous Valet Parking Using Reachable Set Estimation. *Sensors* **2021**, *21*, 1520. <https://doi.org/10.3390/s21041520>

Academic Editors: Francisco J. Martínez and Felipe Jiménez

Received: 28 December 2020
Accepted: 18 February 2021
Published: 22 February 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Autonomous driving technology is being used to support the automotive industry in several ways, ranging from safety concerns to driving comfort. Advances in technology have eliminated drivers' need to pay attention to keeping to driving lanes or maintaining the distance between cars while driving. However, the convenience of automation is limited by the need to park upon the completion of driving, which distresses drivers both mentally and physically [1]. Unfortunately, 23% of all traffic accidents occur in parking lots (car to car collision and car to pedestrian collision), of which 30% are due to parking in an occluded area, where serious injury and damage can occur [2]. If parking can be conducted automatically without human intervention, such a system has the potential to make the driver more comfortable and safer [3].

In this sense, automated valet parking (AVP) systems are one of the most promising technologies for enabling drivers to free themselves from the burden of parking. In 2003, the first automated parking system was introduced into the automotive market [4], which could steer itself into a parking spot. Currently, this initial system has been extended to automated valet parking (AVP), which allows a driver to call a car by pressing a button or instructs the car to park on its own. In the level of automation currently available; however, the driver must continuously monitor the system due to safety concerns.

The main reason for monitoring the automation system is due to the limited sensor range based on the measurement principle, adverse environmental conditions, or occlu-

sions. In addition, for safety reasons, the current field of view must be taken into account, as well as to ensure comfort and to avoid unexpected and harsh reactions. Unfortunately, due to parked vehicles and other structures, the current field of view for sensors in a parking lot is not sufficient for considering new obstacles (pedestrians) coming out of occluded areas. In many approaches of dealing with limited sensor range, new obstacles are treated by reactive planning [5–11], which only deals with visible obstacles. However, if the visibility is limited, proactive planning is required, to detect risks inside an otherwise invisible area.

To reflect the invisible area, we propose a proactive approach to overcome the limited visibility encountered in a parking lot. We divided the proposed approach into three steps: Potential Collision Boundary Estimation, Reachable Set Estimation, and Planning in Limited Visibility. From the first to second step, we estimate the probable maneuver alternatives of other participants by modeling reachable states limited by physics. Then, we introduce a methodology to stay collision-free while considering an environment that includes both limited visibility and possible unexpected behaviors in the last step.

The main contributions are:

1. a deterministic method to represent the risks for unobservable obstacles
2. a framework for generating a safe speed profile in conditions of limited visibility

Previous approaches for the probabilistic modeling of occlusions could not guarantee safe driving; however, we had the planner use the collision risk in an optimized manner by modeling deterministic collision risks in a parking lot. In addition, the problem of occlusion in a parking lot has not yet been covered in other papers. Here, we guarantee the safety issues in conditions of limited visibility by using a deterministic risk estimation process.

The remainder of this paper is structured as follows: In the next section, we shortly review the related work. In Section 3, we start with an overview of the proposed method, before presenting our new approach in Sections 4 and 5. In Section 6, we evaluate the approach in a simulation. Finally, we conclude our work in Section 7.

2. Related Works

Safe motion planning requires the consideration of limited visibility from invisible areas in the environment. Many works have addressed different aspects of risk assessment and safe planning for conditions of limited visibility. Some researchers have studied the probabilistic risk assessment of occluded areas [12–15]. One of them [12] proposed a probabilistic risk assessment method using the volume of traffic on the road, whereas [13] addressed a similar approach using a damage model based on the masses and velocities of two vehicles. However, these algorithms have limitations in that they require the volume of traffic and the mass of the vehicle, which cannot be measured from a sensor. Another method presents the threat level as a probability distribution, using time-to-entry (TTE) and a Bayesian network [14]. In addition, ref. [15] proposed a graphical model capable of describing the risk of a road segment over time, and then addressed the occupancy estimation using a dynamic Bayesian network. Even though these studies explicitly represent the risk as a probability such that the risk can be considered in the planning process, expressing the risk as a probability cannot ensure a provable safety.

To remove the uncertainty of probability, many works use a deterministic approach that solves a worst-case problem [16–22]. The planner presented in [16] dealt with uncertainty predictions at intersections and considers emergency braking before reaching the intersection. The work; however, did not consider sensor range or vehicles approaching behind the perception field. To plan for a fail-safe motion, ref [17] evaluated the occupancy set of vehicles in the environment and considered the existence of an emergency maneuver, e.g., a lane change. This work did not take the perceptive field of the vehicle into account. In another study, ref. [18] presented a method to prevent potentially hazardous situations by with the car cautiously advance into the intersection while regarding possibly occluded traffic participants using a dynamic grid map. Ref. [19] proposed a method to analyze the safety of a given trajectory with respect to occlusions. Ref. [20] focused on motion planning given an uncertain environment model with occlusions. They presented a method for re-

maintaining collision-free for the worst-case evolution of a given scene. Refs. [21,22] formalized the potential risk due to occlusions and limited sensor capability by over-approximating all possible states of unobservable obstacles using state intervals. However, these approaches cannot be applied to valet parking scenarios. Since these algorithms assume that the obstacles pop out according to the topology of the road map, they cannot cautiously assess the spaces between parked vehicles.

3. Overview

The overall process is described in Figure 1a–c. We assume that a path is generated by a priori planner. In other words, we focus on how to consider unobservable obstacles in the speed planning process.

The first step is to estimate a potential collision boundary, which is a boundary from which unobservable obstacles may pop out from, as shown in Figure 1a. By comparing the range sensor and object data, we calculate the location of the potential collision boundary described as the red lines in Figure 1a. Then, we assume that the unobservable obstacles may pop out from these potential collision boundaries.

Unobservable obstacles can be predicted by using the reachable set over-approximations introduced by [23]. A reachable set refers to a method for calculating the distance an obstacle will reach if it is in the collision area, as shown in Figure 1b. We use a simple constant velocity model to predict the reachable set.

The planner plans the speed profile for the unobservable obstacles by defining the problem in the distance-time domain described in Figure 1c. To define this problem, we calculate the start and end times of intersecting unobservable obstacles while following a predefined path. Then, the planner generates a speed profile by solving the A* search algorithm [24]. The details will be explained in the following sections.

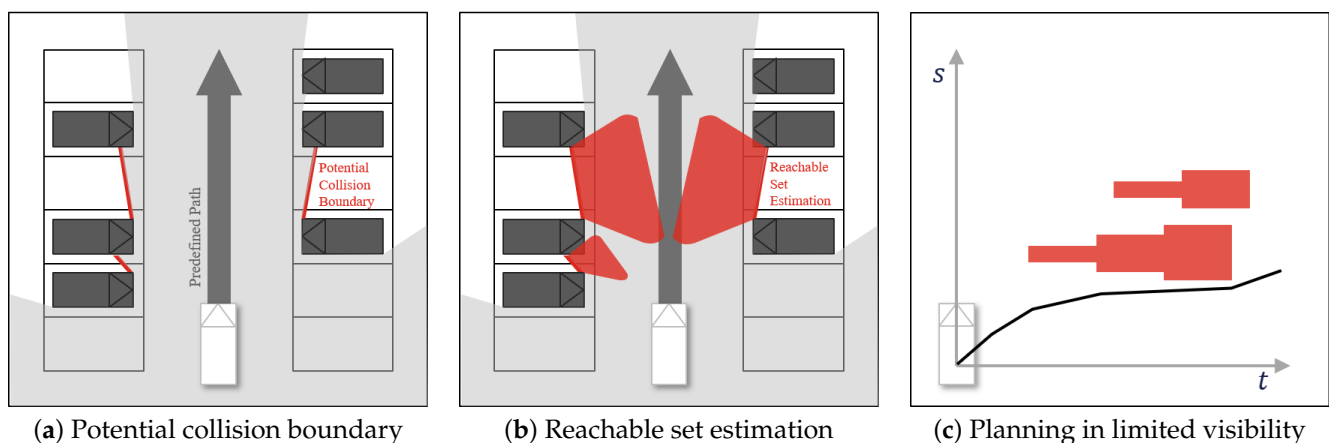


Figure 1. An overall architecture of limited visibility aware motion planning.

4. Risk in Limited Visibility

When the surrounding environment is detected by a sensor mounted on an autonomous vehicle, the field of view might be partially occluded by obstacles such as parked vehicles or constructions. In addition, there may be unobservable pedestrians in the parking lot, which have the potential to collide with the ego vehicle. Therefore, a method for considering occluded areas must be formulated for safe driving. The surrounding environment of the ego vehicle is divided to areas according to whether the vehicle can detect when an obstacle exists. The area surrounded by observable obstacles is known as a “free space”. Conversely, areas with obstacles that interfere with detection of obstacles are called “unknown areas”. A collision boundary is the border between the free space and the unknown area. Among the collision boundaries, the boundary where unobservable obstacles may pop out is called the “potential collision boundary (PCB)”. We assume here

that the unobservable risk from limited visibility originates from the potential collision boundary. The details for calculating PCB will be explained in the following section.

4.1. Estimating the Potential Collision Boundary

To estimate the risk in the occluded boundary, it is first necessary to define a mathematical model for the potential collision boundary (PCB). This model can be derived in three steps. First, candidate points are extracted from the range sensor. Then, based on the extracted candidate points, the free space boundary is formulated as a polygon set. Finally, by subtracting points from observable obstacles at the free space boundary, the PCB can be defined.

In this paper, we use LiDAR sensor to detect the surrounding environment of the ego vehicle. LiDAR provides a point cloud as raw data. Since the point cloud follows the ego vehicle's field of view (FoV), there is no obstacle between the ego vehicle and the point cloud. Therefore, an area consisting of lines that connect each point cloud can be considered a free space, as shown by the gray area in Figure 2.

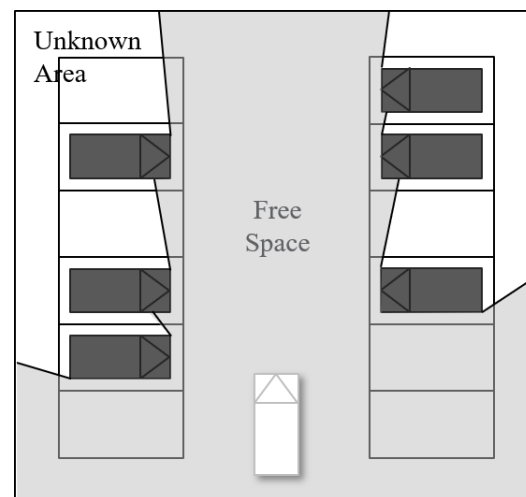


Figure 2. Area classified by the occlusion boundary.

The set of free space polygon is approximated using a line segmentation method. First, the cloud points are arranged in the order of vehicle center and angle, as shown in Figure 3a. Then, we generate a straight line that passes through the two points with the biggest difference in angle, as shown in Figure 3b. Next, we calculate the distances between the straight line and the point from the point cloud to find the farthest point. If the farthest distance is larger than a threshold, the method divides the line at that point. This finding and dividing process is repeated until the farthest distance is less than the threshold. Finally, the free space boundary can be formulated as a set of line segments by adding lines connecting the divided points (Figure 3c). A potential collision may occur between the obstacles (points 3 to 4) and between the obstacle and the maximum range of the sensor (points 6 to 7). For this reason, the cloud points for obstacles (points 1 to 6) are subtracted from the free space boundary, except for the first and the last cloud points of the obstacles. Through the above process, the PCB can be obtained as a set of line segments, as shown by the red lines in Figure 3d.

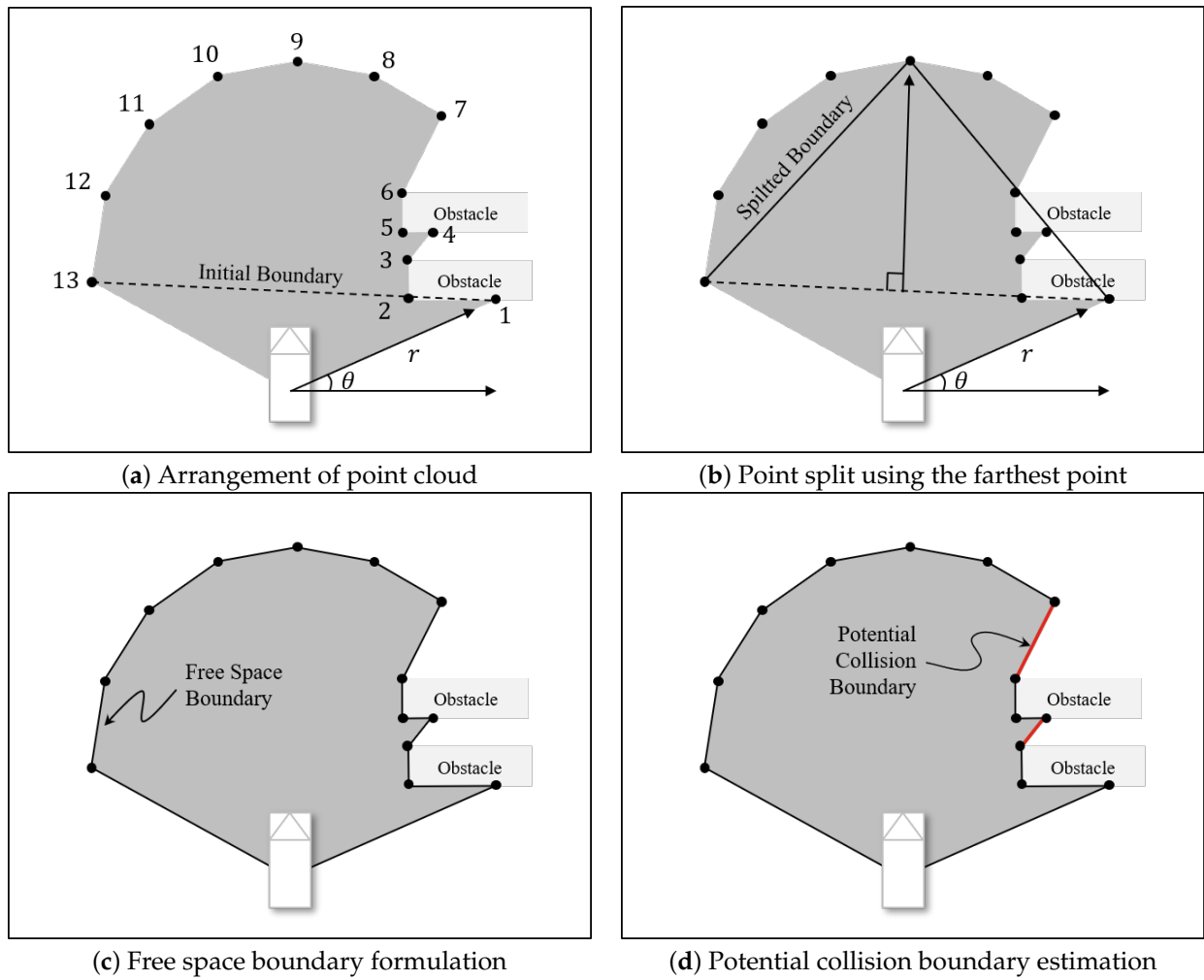


Figure 3. Steps for estimating the potential collision boundary based on candidate points.

4.2. Motion Prediction with Reachable Set Estimation

To estimate the collision risk, we characterize the potential risks from the potential collision boundary. In addition, the ego vehicle and other obstacles is modeled as rectangular shapes, and over-approximations of the unobservable obstacles are modeled using polygons. Here, we define one unobservable obstacle for each potential collision boundary e with the following state, referred as intervals in orientation $\psi_e(0)$, velocity $v_e(0)$, and initial position $s_e(0)$ formulated by the two vertices s_1 and s_2 (see Figure 4). This can be formulated as

$$s_e(0) \in \left[\begin{pmatrix} s_{1,x} \\ s_{1,y} \end{pmatrix}, \begin{pmatrix} s_{2,x} \\ s_{2,y} \end{pmatrix} \right] \quad (1)$$

$$\psi_e(0) \in [\psi_{min}, \psi_{max}] \quad (2)$$

$$v_e(0) \in [v_{min}, v_{max}] \quad (3)$$

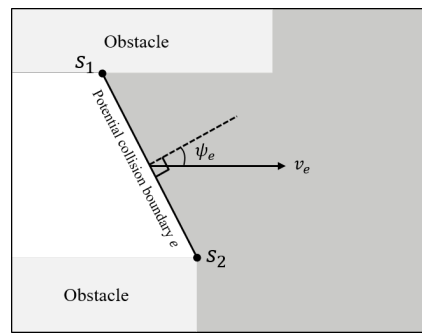


Figure 4. Notations for identifying the potential collision boundary.

The reachable set approximations from [23] for such initial state sets of PCB is derived using the initial state intervals of PCB in Equations (1)–(3). Based on Kamm’s circle [25], these intervals describe the physically reachable area, limited by the absolute possible acceleration. For simplicity, we assume that the state set of PCB can be represented in local coordinates as follows,

$$s_e(0) \in \left[\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \bar{s}_x \\ \bar{s}_y \end{pmatrix} \right] \tag{4}$$

$$\psi_e(0) \in [-\psi_{max}, \psi_{max}] \tag{5}$$

$$v_e(0) \in [v, \bar{v}] \tag{6}$$

Here, we combine a formulation of Kamm’s circle with center $c(t)$ and radius $r(t)$ and the boundary of the circle over time $b(t)$.

$$c(t) = \begin{pmatrix} s_x(0) \\ s_y(0) \end{pmatrix} + \begin{pmatrix} v_x(0) \\ v_y(0) \end{pmatrix} t \tag{7}$$

$$r(t) = \frac{1}{2} a_{max} t^2 \tag{8}$$

$$b_x(t) = v_0 t - \frac{a_{max}^2 t^3}{2v_0} \tag{9}$$

$$b_y(t) = \sqrt{\frac{1}{4} a_{max}^2 t^4 - \left(\frac{a_{max}^2 t^3}{2v_0}\right)^2} \tag{10}$$

Figure 5a–c describes a representation of this estimation.

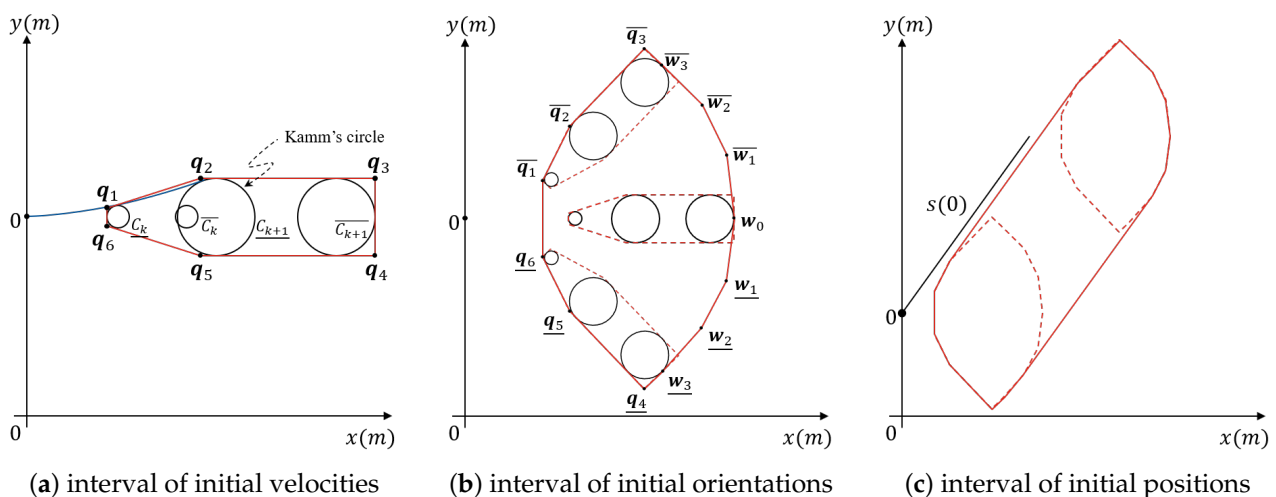


Figure 5. Occupancy over-approximations for initial intervals.

4.2.1. Interval of Initial Velocities

Through the interval of the initial velocity $v_0 \in [\underline{v}, \bar{v}]$ with known orientation $\psi(0) = 0$, and the initial position $s(0) = (0, 0)^T$, we can formulate

$$\underline{c}(t) = c(t, \underline{v}), \bar{c}(t) = c(t, \bar{v}) \quad (11)$$

and likewise, b_x and b_y . Here, $\underline{\cdot}$ is the minimum value of the notation, and $\bar{\cdot}$ refers to the maximum value of the notation. The reachable set of the obstacle for a time period of $\tau_k = [t_k, t_{k+1}]$ can be approximated by the polygon with the points from \mathbf{q}_1 to \mathbf{q}_6 .

$$\mathbf{q}_1 = (c_x(t_k) - r(t_k), r(t_k))^T \quad (12)$$

$$\mathbf{q}_2 = (b_x(t_{k+1}), r(t_{k+1}))^T \quad (13)$$

$$\mathbf{q}_3 = (\bar{c}_x(t_{k+1}) + r(t_{k+1}), r(t_{k+1}))^T \quad (14)$$

$$\mathbf{q}_4 = (\bar{c}_x(t_{k+1}) - r(t_{k+1}), -r(t_{k+1}))^T \quad (15)$$

$$\mathbf{q}_5 = (b_x(t_{k+1}), -r(t_{k+1}))^T \quad (16)$$

$$\mathbf{q}_6 = (c_x(t_k) - r(t_k), -r(t_k))^T \quad (17)$$

as shown in Figure 5a.

The left side of the equation is coincidence with \mathcal{O}_1 , which is the red polygon in Figure 5a, but \mathbf{q}_3 and \mathbf{q}_4 are estimated by using \bar{v} . This boundary includes all $v_i \in [\underline{v}, \bar{v}]$, and each circle $\mathcal{C}_{k+1}(v_i)$ has the same radius $r(t_{k+1})$. Circle center is bounded as $c_x(t_{k+1} \in [\underline{c}_x(t_{k+1}), \bar{c}_x(t_{k+1})], c_y = 0$. Therefore, the polygon $P(\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, \mathbf{q}_4, \mathbf{q}_5, \mathbf{q}_6)$ spanned by $\mathcal{C}_k, \mathcal{C}_{k+1}$, and \mathcal{C}_{k+1} is equivalent to \mathcal{O}_1 . This polygon contains all $\mathcal{C}_t(v_i)$ with $t \in [t_k, t_{k+1}]$, proving that this polygon is an over-approximation of all sets that can be reached for unobservable obstacles with initial velocities

4.2.2. Interval of Initial Orientations

Initial orientation interval $\psi(0) \in [-\psi_{max}, \psi_{max}]$ rotates the entire reachable set $P(\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, \mathbf{q}_4, \mathbf{q}_5, \mathbf{q}_6)$. We can over-approximate this rotated set. The boundaries of the set is formulated by rotating $\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3$ counterclockwise to $\bar{\mathbf{q}}_1, \bar{\mathbf{q}}_2, \bar{\mathbf{q}}_3$ and $\mathbf{q}_4, \mathbf{q}_5, \mathbf{q}_6$ clockwise to $\underline{\mathbf{q}}_4, \underline{\mathbf{q}}_5, \underline{\mathbf{q}}_6$ using ψ_{max} . Furthermore, the furthest longitudinal point $\mathbf{p}_{long} = (\bar{c}_x(k+1) + r(t_{k+1}), 0)^T$ of each circle can be over-approximated by

$$w_0 = \left(\frac{\bar{c}_x + r(t_{k+1})}{\cos \frac{\theta}{2}} \right)^T, \theta = \frac{\psi_{max}}{n} \quad (18)$$

$$\bar{W}_j = R_{\theta_j} w_0 \quad (19)$$

$$\underline{W}_j = -R_{\theta_j} w_0 \quad (20)$$

with $j \in [1, n]$. An example approximation of the circle with ψ_{max} of around 45 deg is achieved with $n = 3$. Figure 5b describes this formula, proving that each rotated polygon over $[-\psi_{max}, \psi_{max}]$ is included by the following polygon.

$$P(\bar{\mathbf{q}}_1, \bar{\mathbf{q}}_2, \bar{\mathbf{q}}_3, \bar{w}_n, \dots, \bar{w}_1, w_0, w_n, \underline{\mathbf{q}}_4, \underline{\mathbf{q}}_5, \underline{\mathbf{q}}_6) \quad (21)$$

4.2.3. Interval of Initial Positions

The transform due to intervals of initial position is defined by using linear interpolation of the polygon \underline{P} for $s(0) = (0,0)^T$ as described in the previous subsection. First, we create a duplicate \bar{P} that is translated by $\bar{s}(0) = (s_x, s_y)^T$ and then compute the total polygon of both polygons $\mathcal{O}_1(\tau_k) = \text{Conv}(P, \bar{P})$. The occupancy of each possible position on the line segment is easily calculated by the linearity of translations and the line segments. Thus, $\mathcal{O}_1(\tau_k)$ is an over-approximation of the possible reachable set of an unobservable obstacle with a bounded initial state. The resulting over-approximation is described with example parameters in Figure 5c.

The total occupancy \mathcal{O}_1 refers to the reachable area if the unobservable obstacle originates from the potential collision boundary e . By applying the reachable set estimation, we can calculate the intersection area between the reachable set and the predefined path. In the planning phase, we generate an optimal speed profile to consider for the intersection area. Further details will be explained in the following chapter.

5. Planning in Limited Visibility

Prior to planning the speed profile for conditions of limited visibility, the path should be defined. Since most situations can be formulated as driving along a predefined lane with arbitrary geometry, we assume that the problem of roaming in a parking lot is similar to the problem of driving along a lane. For this reason, we use the trajectory planner defined in [10] to generate a path in a parking lot. In the speed planner, we only calculate in the longitudinal direction, i.e., we set the problem in a one-dimensional direction. This approach is known as a path-velocity decomposition [26].

We first construct the planning environments for the problem as written in [27]. This approach provides a general solution for a longitudinal direction in on-road driving if the planning environments can be described. The planning environments include how to obtain the desired speed, how to represent obstacles, and how to set a goal. Then, the planner solves the optimization problem using A^* , which is a well-known planning algorithm.

The two main differences with the previous approach are:

1. the proposed approach considers not only observable obstacles but also unobservable obstacles
2. the proposed approach does not require a topology map

We predict the motion of unobservable obstacles using the over-approximation of PCBs in the previous section. The PCBs are over-approximated by estimating all possible motions when unobservable obstacles suddenly come out from the PCBs, as shown in Figure 5. This over-approximated area covers the position where unobservable obstacles pop out. Therefore, through the over-approximation, unobservable obstacles can be represented as with observability.

In addition, the previous approach can only represent obstacles on the road topology as the planning environments. This is appropriate to on-road planning, but not in parking lots since unobservable obstacles can be a pedestrian and a cyclist who does not follow the road topology. However, the proposed approach can deal with these obstacles. The PCBs are calculated from LiDAR sensor field of view, which is not associated with the road topology, then, the over-approximation of PCBs can cover the unpredictable motion of unobservable obstacles such as a pedestrian or a cyclist.

5.1. Problem Statement

Assume that $p_i = (p_x, p_y)^T \in \mathbb{R}^2$ is a point on the center line of a predefined path c , then $s(p_i) \in \mathbb{R}$ denotes the traveled distance along the path in the interval $[p_0, p_i]$. The velocity is bounded as $[0, v_{max}]$, and then, $v_{max}(s)$ is a function of the path's curvature κ at distance s , i.e., $v_{max}(s) = f(\kappa(s))$. u , the acceleration of the vehicle, is the system

input within $[a_{min}, a_{max}]$. The longitudinal movement of the vehicle is formulated by the differential equations as follows,

$$\begin{bmatrix} \dot{s} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} s \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \quad (22)$$

Along the path c , there is a finite set E of obstacles E_i that intersects with the path for a time period $\tau^{E_i} = [t_{start}^{E_i}, t_{end}^{E_i}]$ at a specific position $s^{E_i}(t)$ and time τ^{E_i} . These obstacles should not occupy the position of the ego vehicle s_{ego} at any time. The goal of the planner is to find a valid speed profile. This plan can be derived as an optimization problem over f , such that

$$\min_{u(t)} (f) = \min_{u(t)} f(s_{ego}, \dot{s}_{ego}, \dot{v}_{ego}, \kappa(s), E) \quad (23)$$

This problem must have only one global minimum for various constraints. The optimization problem can be converted to a discrete problem in the state space $\mathcal{X} \subseteq \mathbb{R}^3$ with states $x = [s, v, t]^T \in \mathcal{X}$. Then, this problem can be solved using an A* graph search [24]. The state x_i denotes the state at the planning step i . We construct the searching graph for A* online by sampling a set of actions \mathcal{A} during a time step Δt . At each iteration of its main loop, A* needs to determine which of its paths to extend. It does so based on the cost of the path, using an estimate of the cost required to extend the path to the goal. Specifically, A* selects the path that minimizes

$$f(x_i) = g(x_i) + h(x_i) \quad (24)$$

where $g(\cdot)$ is the cost of the path from the initial state, and $h(\cdot)$ is a heuristic function that estimates the cost for the cheapest path from the current state to the goal. A* terminates when the path it chooses to extend is a path from start to goal, or if there are no paths eligible to be extended. The heuristic function is problem-specific. If the heuristic function is admissible, meaning that it never overestimates the actual cost to get to the goal, A* is guaranteed to return the least-cost path from the start to the goal. The following sections describe how the A* graph is constructed, its cost function, and heuristics.

5.2. Transition Model

The discretized transition model can be written as

$$x_{i+1} = \begin{bmatrix} s_{i+1} \\ v_{i+1} \\ t_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \end{bmatrix} \begin{bmatrix} s_i \\ v_i \\ t_i \\ 1 \end{bmatrix} + \begin{bmatrix} \frac{1}{2}(\Delta t)^2 \\ \Delta t \\ 0 \end{bmatrix} a_i \quad (25)$$

where a_i refers the action that is selected in the step i and expanded for Δt . (25) represents the longitudinal motion of the ego vehicle. The objective of planning is to find a set of x_i in an acyclic searching graph. To reduce the computational time, we discretize the action space. By using a discretized action space, the state x_i is expanded to reach a goal state x_G . At that time, the searching graph is generated, then, we select the minimum cost state from among the next states. This process repeats until the state x_i reaches x_G .

5.3. Cost Function

$g(x_i, a, x_{i+1}, E)$ is the step cost in state x_i to traverse to state x_{i+1} for taking action a . The total cost is the cumulative cost of all steps along the path, $\sum_{x_i=x_{start}}^{x_{goal}} g(x_i, a, x_{i+1}, E)$. The goal is to find the path from the initial state to a goal state that incurs minimal costs. To represent the different parameters of the optimization problem, a weighted sum of different costs is used as the stop cost.

$$g(x_i, a, x_{i+1}, E) = \omega_V \cdot g_V(s_{i+1}) + \omega_A \cdot g_A(a) + \omega_E \cdot g_E(x_i, x_{i+1}, E) \quad (26)$$

where $g_V(x_{i+1})$ refers to the cost for the desired speed, $g_A(a)$ denotes the cost of taking action a , and $g_E(x_i, x_{i+1}, E)$ is the cost for a collision during traveling from x_i to x_{i+1} . In addition, ω_V, ω_A , and ω_E are the weight factors for the desired speed, acceleration, and collision, respectively. The following sections derive the formulation for each cost function.

5.3.1. Velocity Cost

$v_{des}(s)$ is a function of the desired speed at position s without obstacles. This speed combines the speed limit $v_{law}(s)$ and $v_{curve}(s)$ along the path length s . v_{curve} is formulated according to a maximum allowed lateral acceleration $a_{lat,curve}$ in the curvature radius $r_{curve}(s)$ written in [28].

$$v_{curve}(s) = \sqrt{a_{lat,curve} r_{curve}(s)} \quad (27)$$

Then, the desired speed is the minimum of v_{law} and v_{curve} as shown in Figure 6. We set the speed limit in a parking lot to be under 15 km/h, and the maximum allowed lateral acceleration is 2 m/s².

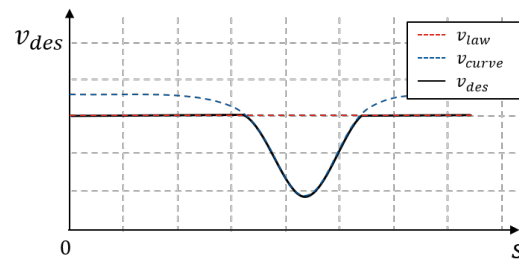


Figure 6. Along the path, the velocity is limited by the speed limit and the curvature. The curve approach velocity is part of the desired speed.

The velocity cost g_V is derived as difference to the desired speed v_{des} . A too high speed is punished quadratically, while a too low speed is punished linearly to allow for a lower speed when decelerating upon obstacles. Here, $g_V(x_{i+1})$ can then be written as follows:

$$g_V(x_{i+1}) = \begin{cases} (v_{i+1} - v_{des}(s_{i+1}))^2, & v_{i+1} > v_{des}(s_{i+1}) \\ 0, & v_{i+1} = v_{des}(s_{i+1}) \\ \frac{1}{2}(v_{des}(s_{i+1}) - v_{i+1})^2, & v_{i+1} < v_{des}(s_{i+1}) \end{cases} \quad (28)$$

5.3.2. Acceleration Cost

The cost of taking action depends on the acceleration value a . The objective of the acceleration cost is to punish either a too high or too low acceleration, and to maintain the current acceleration. Here, $g_A(a)$ is written as follows:

$$g_A(a) = \frac{1}{2}a^2 \quad (29)$$

The use of an acceleration cost or a high value of acceleration weighting factor allows for comfortable driving.

5.3.3. Collision Cost

The ego vehicle may be encounter observable and unobservable obstacles such as vehicles and pedestrians. Therefore, the idea is to construct a simple representation for obstacles, which includes the representation of any possible occurrence in the longitudinal direction. An obstacle E_i intersects the path at position $s(t)$ at time interval $\tau \in [t_{start}, t_{end}]$, then the obstacle E_i has a length l^{E_i} . The obstacle also has a desired following distance $d_{des}^{E_i}$, which is defined as a temporal-spatial cost map M^{E_i} required to achieve a smooth driving. The cost map M^{E_i} is a linear function using the obstacle representation of $E_i =$

$(s^{E_i}(t), d_{des}^{E_i}, l^{E_i}, t_{start}, t_{end})$. This map is illustrated in Figure 7. The collision cost can be derived as

$$g_E(x_i, x_{i+1}, E) = \begin{cases} \infty, & \text{if } \exists E_i \in E : x_{i+1} \in E_i \\ M^{E_i}(x_{i+1}), & \text{if } \exists E_i \in E : x_{i+1} \in E_i \\ 0, & \text{otherwise} \end{cases} \quad (30)$$

By applying the reachable set estimation, unobservable obstacles can be regarded as observable obstacles when defined as E_i . Through the time intersecting the reachable set and the predefined path, we can define $t_{start}^{E_i}$ and $t_{end}^{E_i}$, and the intersecting distance then describes the position of the obstacle. The desired following distance for the unobservable obstacle can subsequently be calculated in the same manner as for the observable obstacle.

$$d_{des}^{E_i} = d_{threshold} + v_{ego} \cdot t_{gap} \quad (31)$$

where $d_{threshold}$ refers to the ideal distance at zero speed, v_{ego} is the current velocity of the ego vehicle, and t_{gap} denotes the ideal time gap for the front obstacle. The linear function for the cost map M^{E_i} can then be written as

$$M^{E_i} = \begin{cases} d_{des}^{E_i} - d^{E_i}, & \text{if } d^{E_i} < d_{des}^{E_i} \\ 0, & \text{otherwise} \end{cases} \quad (32)$$

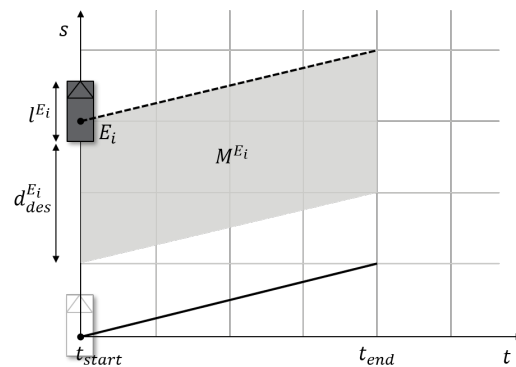


Figure 7. Representation of obstacles in the distance-time domain.

5.4. Heuristics

An appropriate and consistent heuristic cost can reduce the computation of A* algorithm. Here, we use Inevitable Collision States (ICSs) [29] as a heuristic cost. An ICS is a state that cannot avoid at least one collision in the future. When a new state is expanded, this state is tested for whether the new state is an ICS. If this is true, i.e., the new state cannot avoid a collision, the remaining heuristic cost h_{x_i} is the maximum collision cost. Through the ICS, the planner enables admissible reactions to upcoming obstacles. In the case of a movement in a one-dimensional direction, the ICS test can be easily done analytically. The ICS test for newly generated state x_i can be derived as

$$\forall a \in \mathcal{A} \exists E_i \in E : \{s_i + v_i \cdot t + \frac{1}{2}a \cdot t^2 | t \in [0, \infty)\} \neq \emptyset \quad (33)$$

Figure 8 presents the concept of an ICS. There are two different initial states x_1 and x_2 , with $v_1 < v_2$. A collision cannot be avoided for v_2 , whereas the vehicle can avoid the obstacle with v_1 . The heuristic function can be written as

$$h(x_i) = \begin{cases} \infty, & \text{if } x_i \in \text{ICS} \\ 0, & \text{otherwise} \end{cases} \quad (34)$$

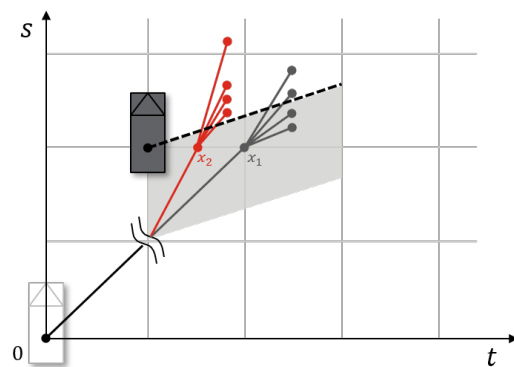


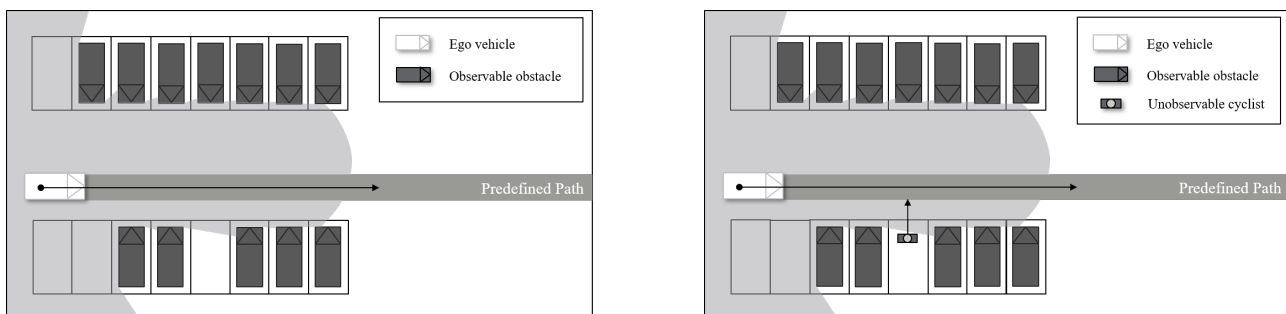
Figure 8. Analytic calculation of the Inevitable Collision State.

5.5. Goals

The original A* algorithm has a specific goal in the constructed graph. Since we build the graph online, the goal may or may not be reached. Therefore, we set a partial goal in the time domain t_G . No matter where the vehicle is or its speed, the planner stops finding the speed profile when the time of state t_i reaches t_G .

6. Simulation Results

We evaluated the proposed algorithm using two scenarios, as shown in Figure 9. The first scenario includes the ego vehicle roaming in a parking lot where parked vehicles exist. From this scenario, the proposed method shows the ability to drive safely in limited visibility. The second scenario includes a cyclist that pops out from the potential collision boundary. Through the second scenario, the safety of the proposed algorithm will be proved.



(a) Roaming in a parking lot without moving obstacles **(b)** a cyclist pops out from the potential collision boundary

Figure 9. Scenario descriptions.

The tests are conducted using a Robot Operating System (ROS) platform [30] in a CARLA simulator [31]. The proposed algorithm was implemented using an i5 Core PC in C++ language. The execution periods of the overall planning process were 100 ms. The total planning horizon was 5 s, and the time step was set to 0.5 s.

6.1. Scenario 1: Roaming in a Parking Lot without Obstacles

As first online scenario, roaming in a parking lot is presented. The ego vehicle roams a parking lot with many parked vehicles. The ego vehicle must consider unobservable obstacles that suddenly pops out between the parked vehicles. In other words, the ego vehicle should consider the uncertain prediction of unobservable obstacles which is realized by the reachable set estimation. This uncertainty is incorporated by estimating reachable set in Section 4.

In Figure 10a, as a control with the proposed algorithm, the ego vehicle can be seen driving in a parking lot without other parked vehicles. On the other hand, Figure 10 shows

the process of roaming in a parking lot. At each figure of Figure 10b–d, the upper figure shows the ego vehicle’s front camera image. The middle figure describes the algorithm details in ROS platform. In this figure, the green points are point cloud and the red lines refers the potential collision boundaries (PCBs). The lower figure shows the estimated reachable set as the red polygon and the speed profile generated by the proposed algorithm.

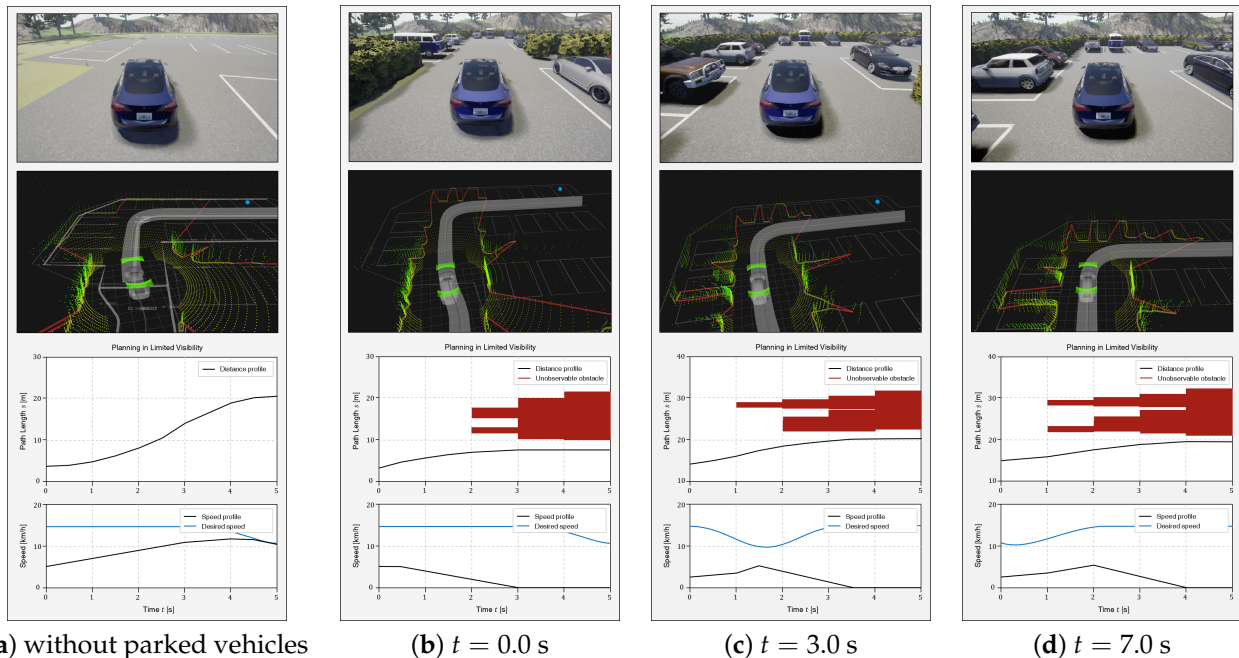


Figure 10. Roaming in a parking lot without obstacles. The upper figure of each time step shows the situation in CARLA. The second figure describes the algorithm in ROS platform. The last figure shows the speed profile at each time step.

The ego vehicle generates the speed profile to follow the regulation speed described as the blue line in the lower figure in Figure 10a if the risk of limited visibility does not exist. On the contrary, at $t = 0$ (see Figure 10b), the ego vehicle moves toward an empty parking space without observable obstacles. The ego vehicle tries to generate a speed profile to keep up the regulation speed. However, the ego vehicle cannot speed up until it reaches the regulation speed because it collides with invisible obstacles when speeding up. At that time, the prediction of unobservable obstacles is realized by reachable set estimation as shown in the red polygon in the lower figure of Figure 10b–d. The reachable set is transformed into the distance-time domain with the red polygon in the lower figure. Then, the ego vehicle continuously decelerates until the field of view is large enough to eliminate PCBs. Since the parking lot is not enough to secure visibility due to parked vehicles, the ego vehicle maintains a low speed than the regulation during roaming. (see Figure 10b–d).

6.2. Scenario 2: Roaming in a Parking Lot with Obstacles

Same as the first scenario, the ego vehicle roams a parking lot with many parked vehicles. The difference is that a cyclist suddenly pops out from behind a parked vehicle after few second. The ego vehicle cannot detect the cyclist due to the occlusion. This scenario evaluates the model capability when an obstacle pops out from the PCB. In this scenario, an unobservable obstacle is detected at the PCB, at which time it is converted into an observable obstacle. Consequently, the proposed algorithm highlights its scalability for all types of obstacles (unobservable and observable).

Initially, the ego vehicle drives in a parking lot as shown in Figure 11a. Similar to the above simulation, the proposed algorithm tries to decrease the speed slower than the regulation. At the same time, a cyclist is behind the parked vehicle; however, the ego vehicle cannot detect the occluded cyclist. Even though the ego vehicle suddenly encounters the cyclist at $t = 2$ as shown in Figure 11b, the proposed algorithm can stop

to avoid a collision with the cyclist since the speed is slow enough as it nears the PCB. The cyclist is described as the yellow polygon in the Figure 11b. The proposed algorithm waits for the cyclist to pass, then increases the speed to arrive at the empty parking space, as shown in Figure 11c. This simulation results confirm that the reachable set estimation is well defined to consider unobservable obstacles occluded by other traffic participants. Moreover, the cost and heuristic functions of the guided A* algorithm are guaranteed to consider both observable and unobservable obstacles.

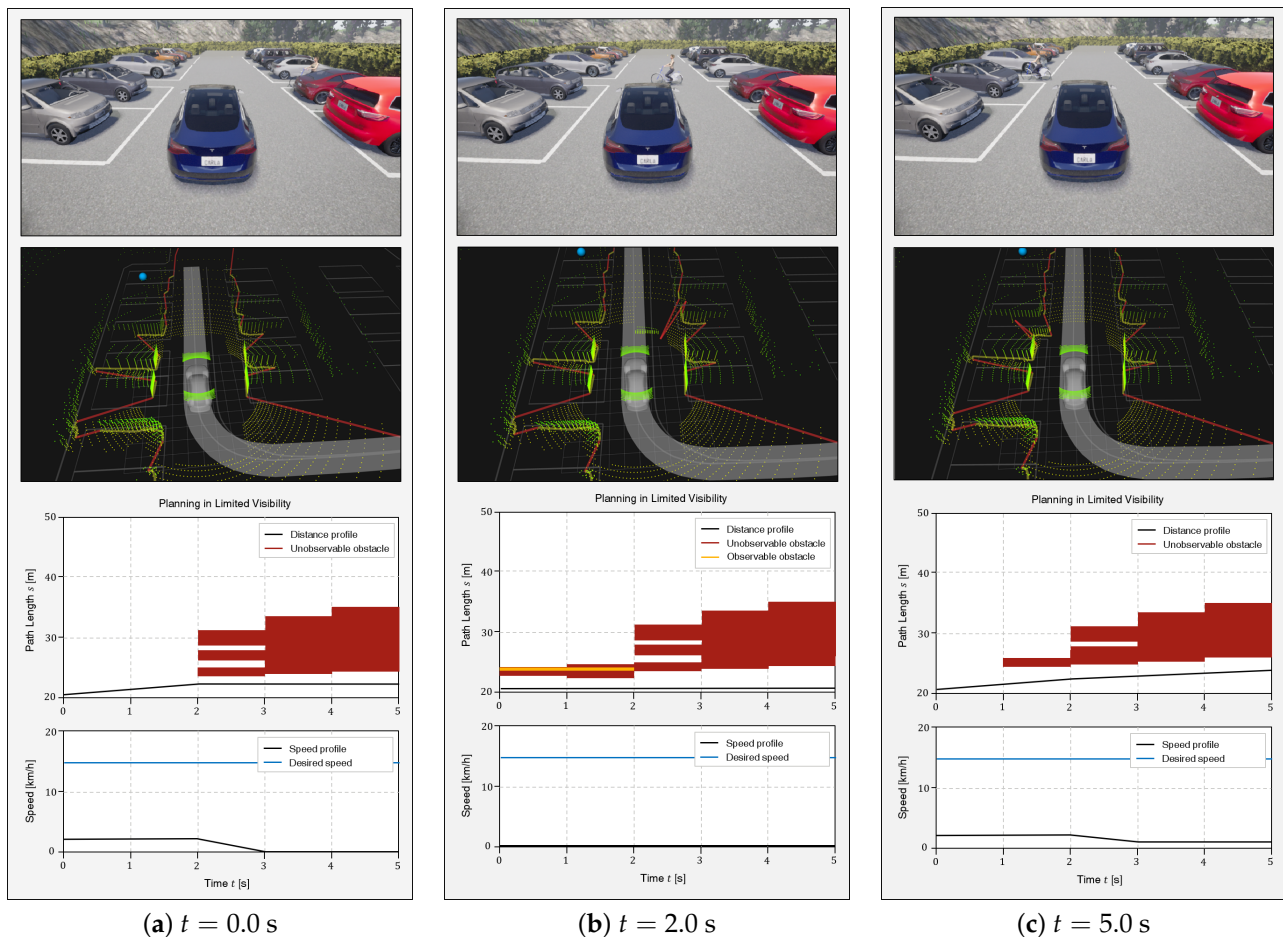


Figure 11. Roaming in a parking lot with obstacles. The upper figure of each time step shows the situation in CARLA. The second figure describes the algorithm in ROS platform. The last figure shows the speed profile at each time step.

7. Conclusions

This paper presented an algorithm for safe motion planning in condition of limited visibility in a parking lot. The proposed algorithm consists of three steps: Potential Collision Boundary (PCB) Estimation, Reachable Set Estimation-based Motion Prediction, and Motion Planning with A*. To consider visibility limited by occlusions, we defined a PCB to describe the risk of unobservable obstacles. By applying a reachable set estimation as the motion prediction, the PCB is extended to the reachable area once it is determined that an unobservable obstacle exists. Through over-approximations, the reachable set estimation represents the worst-case of unobservable obstacles. We then rewrite the reachable set to include a state for the planning space, such that a simple representation model can convert the reachable set to obstacles in the A* algorithm. In this way, we use the A* algorithm as a motion planner to achieve completeness, optimality, and optimal efficiency. Using this algorithm, we can formulate the cost function to ensure the vehicle follows the desired speed, ensure the vehicle is driving comfortably, and that it avoids collisions. In

addition, the Inevitable Collision State (ICS) is applied to the heuristic function to reduce the computational burden.

The simulation results show the ability of the algorithm to generate an optimal speed profile for unobservable obstacles. Through the proposed algorithm, the vehicle could drive in an occluded area with no collisions. Since we enhanced the reachable set by using over-approximations, we could confirm that the planner planned safe speed profiles. The performance is shown via roaming scenarios with occlusions from parked vehicles. All collisions could be prevented while still moving through the parking lot at a sufficient speed. The proposed planner could proactively adjust to unobservable obstacles using one algorithm in an optimized manner, by solving the optimization problem.

Nevertheless, future work will be extended in two ways. On the first hand, the idea is to apply to real driving. The most important thing to apply a planning algorithm to real driving is to handle the uncertainty of input, in this case, the point cloud. If the point cloud measurement is noisy, it causes both the free space and the PCB are also noisy. Then, the planner might mispredict the reachable set of potential collision obstacles. Thus, the speed profile can be noisy in real driving. To overcome this problem, we should develop a PCB tracking algorithm or a fusion algorithm between free spaces from LiDAR and from the camera. Second, we proposed a motion planner that only considers a longitudinal direction. In other words, we solved the problem by either reducing or increasing the speed. However, in more complex situations, such as crossing an intersection in a parking lot, waiting for a parked vehicle, and encountering a car in a narrow alleyway, the planner must change paths laterally. Therefore, we should consider both longitudinal and lateral directions of the trajectory. To consider a lateral direction; however, the complex situation will be divided into a normal state and an inevitable state. A normal state refers to the problem described in Section 3, whereas an inevitable state indicated the complicated situation mentioned above. The different methods should be applied to different types of problems.

Author Contributions: Conceptualization, S.L., W.L., M.S. and K.J.; methodology, S.L. and W.L.; software, S.L.; validation, S.L.; formal analysis, S.L.; investigation, S.L., W.L., M.S. and K.J.; resources, S.L.; data curation, S.L.; writing—original draft preparation, S.L., W.L., M.S. and K.J.; writing—review and editing, S.L.; visualization, S.L.; supervision, S.L.; project administration, K.J.; funding acquisition, M.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was financially supported by the Industrial Strategy Technology Development Program (No. 10039673, 10079961), the International Collaborative Research and Development Program (N0001992) under the Ministry of Trade, Industry and Energy (MOTIE Korea), and National Research Foundation of Korea (NRF) grant funded by the Korean government (MEST) (No. 2011-0017495).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing not applicable .

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

References

1. Kotb, A.O.; Shen, Y.C.; Huang, Y. Smart Parking Guidance, Monitoring and Reservations: A Review. *IEEE Intell. Transp. Syst. Mag.* **2017**, *9*, 6–16. [[CrossRef](#)]
2. Avery, M. Automated Driving: The Technology and Implications for Insurance. Available online: <https://etsc.eu/wp-content/uploads/MatthewAvery.pdf> (accessed on 28 December 2020).
3. Banzhaf, H.; Nienhuser, D.; Knoop, S.; Marius Zollner, J. The future of parking: A survey on automated valet parking with an outlook on high density parking. In Proceedings of the 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA, 11–14 June 2017; pp. 1827–1834. [[CrossRef](#)]
4. Kageyama, Y. Look, No Hand! New Toyota Parks Itself. 2004. Available online: <https://www.goupstate.com/article/NC/20040115/news/605146346/SJ> (accessed on 28 December 2020)

5. Werling, M.; Ziegler, J.; Kammel, S.; Thrun, S. Optimal trajectory generation for dynamic street scenarios in a frenét frame. In Proceedings of the 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 3–7 May 2010; pp. 987–993. [\[CrossRef\]](#)
6. Ziegler, J.; Bender, P.; Dang, T.; Stiller, C. Trajectory planning for Bertha—A local, continuous method. In Proceedings of the 2014 IEEE Intelligent Vehicles Symposium Proceedings, Dearborn, MI, USA, 8–11 June 2014; pp. 450–457. [\[CrossRef\]](#)
7. Ziegler, J.; Stiller, C. Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios. In Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, MO, USA, 10–15 October 2009; pp. 1879–1884. [\[CrossRef\]](#)
8. Kuwata, Y.; Fiore, G.A.; Teo, J.; Frazzoli, E.; How, J.P. Motion planning for urban driving using RRT. In Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, France, 22–26 September 2008; pp. 1681–1686. [\[CrossRef\]](#)
9. Kelly, A.; Nagy, B. Reactive Nonholonomic Trajectory Generation via Parametric Optimal Control. *Int. J. Robot. Res.* **2003**, *22*, 583–601. [\[CrossRef\]](#)
10. Lim, W.; Lee, S.; Sunwoo, M.; Jo, K. Hierarchical Trajectory Planning of an Autonomous Car Based on the Integration of a Sampling and an Optimization Method. *IEEE Trans. Intell. Transp. Syst.* **2018**, *19*, 613–626. [\[CrossRef\]](#)
11. Falcone, P.; Borrelli, F.; Asgari, J.; Tseng, H.E.; Hrovat, D. Predictive active steering control for autonomous vehicle systems. *IEEE Trans. Control Syst. Technol.* **2007**, *15*, 566–580. [\[CrossRef\]](#)
12. Lee, M.; Sunwoo, M.; Jo, K. Collision risk assessment of occluded vehicle based on the motion predictions using the precise road map. *Robot. Auton. Syst.* **2018**, *106*, 179–191. [\[CrossRef\]](#)
13. Damerow, F.; Pupal, T.; Li, Y.; Eggert, J. Risk-based driver assistance for approaching intersections of limited visibility. In Proceedings of the 2017 IEEE International Conference on Vehicular Electronics and Safety (ICVES), Vienna, Austria, 27–28 June 2017; pp. 178–184. [\[CrossRef\]](#)
14. Noh, S. Decision-Making Framework for Autonomous Driving at Road Intersections: Safeguarding Against Collision, Overly Conservative Behavior, and Violation Vehicles. *IEEE Trans. Ind. Electron.* **2019**, *66*, 3275–3286. [\[CrossRef\]](#)
15. McGill, S.G.; Rosman, G.; Ort, T.; Pierson, A.; Gilitschenski, I.; Araki, B.; Fletcher, L.; Karaman, S.; Rus, D.; Leonard, J.J. Probabilistic Risk Metrics for Navigating Occluded Intersections. *IEEE Robot. Autom. Lett.* **2019**, *4*, [\[CrossRef\]](#)
16. De Campos, G.R.; Runarsson, A.H.; Granum, F.; Falcone, P.; Alenljung, K. Collision avoidance at intersections: A probabilistic threat-assessment and decision-making system for safety interventions. In Proceedings of the 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), Qingdao, China, 8–11 October 2014; pp. 649–654. [\[CrossRef\]](#)
17. Magdici, S.; Althoff, M. Fail-safe motion planning of autonomous vehicles. In Proceedings of the 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), Rio de Janeiro, Brazil, 1–4 November 2016; pp. 452–458. [\[CrossRef\]](#)
18. Hoermann, S.; Stumper, D.; Dietmayer, K. Probabilistic long-Term prediction for autonomous vehicles. In Proceedings of the 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA, 11–14 June 2017; pp. 237–243. [\[CrossRef\]](#)
19. Naumann, M.; Königshof, H.; Lauer, M.; Stiller, C. Safe but not overcautious motion planning under occlusions and limited sensor range. In Proceedings of the 2019 IEEE Intelligent Vehicles Symposium (IV), Paris, France, 9–12 June 2019; pp. 140–145. [\[CrossRef\]](#)
20. Tas, O.S.; Hauser, F.; Stiller, C. Decision- Time Postponing Motion Planning for Combinatorial Uncertain Maneuvering. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 2419–2425. [\[CrossRef\]](#)
21. Orzechowski, P.F.; Meyer, A.; Lauer, M. Tackling Occlusions Limited Sensor Range with Set-based Safety Verification. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 1729–1736. [\[CrossRef\]](#)
22. Orzechowski, P.F.; Li, K.; Lauer, M. Towards responsibility-sensitive safety of automated vehicles with reachable set analysis. In Proceedings of the 2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE), Graz, Austria, 4–8 November 2019. [\[CrossRef\]](#)
23. Althoff, M.; Magdici, S. Set-based prediction of traffic participants on arbitrary road networks. *IEEE Trans. Intell. Veh.* **2016**, *1*, 187–202. [\[CrossRef\]](#)
24. Russell, S.; Norving, P. *Artificial Intelligence: A Modern Approach*, 4th ed.; Prentice Hall Press: Upper Saddle River, NJ, USA, 2020.
25. Rajamani, R. *Vehicle Dynamics and Control*; Mechanical Engineering Series; Springer: Boston, MA, USA, 2012. [\[CrossRef\]](#)
26. Kant, K.; Zucker, S.W. Toward Efficient Trajectory Planning: The Path-Velocity Decomposition. *Int. J. Robot. Res.* **1986**, *5*, 72–89. [\[CrossRef\]](#)
27. Hubmann, C.; Aeberhard, M.; Stiller, C. A generic driving strategy for urban environments. In Proceedings of the 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), Rio de Janeiro, Brazil, 1–4 November 2016; pp. 1010–1016. [\[CrossRef\]](#)
28. Kohlhaas, R.; Schamm, T.; Lenk, D.; Zollner, J.M. Towards driving autonomously: Autonomous cruise control in urban environments. In Proceedings of the 2013 IEEE Intelligent Vehicles Symposium (IV), Gold Coast, Australia, 23–26 June 2013; pp. 109–114. [\[CrossRef\]](#)
29. Fraichard, T.; Asama, H. Inevitable collision states—A step towards safer robots. *Adv. Robot. Taylor Fr. Group* **2004**, *18*, 1001–1024. [\[CrossRef\]](#)

-
30. Quigley, M.; Gerkey, B.; Conley, K.; Faust, J.; Foote, T.; Leibs, J.; Berger, E.; Wheeler, R.; Ng, A. ROS: An open-source Robot Operating System. In Proceedings of the ICRA Workshop on Open Source Software, Kobe, Japan, 12–13 May 2009.
 31. Dosovitskiy, A.; Ros, G.; Codevilla, F.; Lopez, A.; Koltun, V. CARLA: An Open Urban Driving Simulator. 2017. Available online: <http://xxx.lanl.gov/abs/1711.03938> (accessed on 28 December 2020).