Foresee the Unseen: Sequential Reasoning about Hidden Obstacles for Safe Driving

José Manuel Gaspar Sánchez*, Truls Nyberg†, Christian Pek, Jana Tumova and Martin Törngren

Abstract—Safe driving requires autonomous vehicles to anticipate potential hidden traffic participants and other unseen objects, such as a cyclist hidden behind a large vehicle, or an object on the road hidden behind a building. Existing methods are usually unable to consider all possible shapes and orientations of such obstacles. They also typically do not reason about observations of hidden obstacles over time, leading to conservative anticipations. We overcome these limitations by (1) modeling possible hidden obstacles as a set of states of a point mass model and (2) sequential reasoning based on reachability analysis and previous observations. Based on (1), our method is safer, since we anticipate obstacles of arbitrary unknown shapes and orientations. In addition, (2) increases the available drivable space when planning trajectories for autonomous vehicles. In our experiments, we demonstrate that our method, at no expense of safety, gives rise to significant reductions in time to traverse various intersection scenarios from the CommonRoad Benchmark Suite.

I. INTRODUCTION

As for any driver, autonomous vehicles (AVs) should strive for defensive driving strategies in order to ensure safety. This means that AVs must be cautious in challenging situations, such as the intersection depicted in Fig. 1.a. In this example, the view of the blue AV in the bottom (referred to in Fig. 1 as ego vehicle) is blocked by the yellow vehicle turning right (referred to in Fig. 1 as Obstacle). The AV needs to consider possible hidden obstacles in the unseen region to plan a safe trajectory through the intersection and avoid a collision with the hidden motorcycle.

A challenge for the AV is, however, not to drive too defensively either. To avoid hindering the AVs’ progress, assumptions on unseen traffic participants may be needed, such that they adhere to traffic rules and speed limits (to some extent). Under the premises that these assumptions hold, it is possible to guarantee legal safety, i.e. that the AV will not plan a maneuver that causes an accident [1].

Ideally though, the AV should assume as little as possible about other traffic participants’ cooperation. It should allow for reasonable human errors, such as long reaction times or minor traffic rule infringements.

In Fig. 1.b, the ego vehicle aims to turn left in an intersection. Through previous observations (of the free lane on the top and the translucent yellow car on the left), it should be able to conclude that no obstacle is approaching from the right hidden behind the yellow vehicle. To the best of our knowledge, no previously proposed method demonstrates that utilizing past observations can alleviate the problem of the AV being too conservative when considering possible hidden obstacles.

A. Related Work

Problems with limited visibility are addressed in several recent works. Indoor robots are considered in [2], where the authors propose a method for path planning based on a preprocessing step. Due to its offline nature, this method does not take into account hidden objects caused by other dynamic obstacles.
The more general case of a mobile robot navigating in an unknown dynamic environment is analyzed in [3]. In their work, the authors shrink the drivable area based on an assumed maximum speed of the unknown moving objects. This approach guarantees that, if a collision occurs, the ego robot will be standing still.

Also in the field of autonomous driving, several works have recently focused on tackling limited view and hidden obstacles. A probabilistic approach for computing the risk of collision with an unseen vehicle in an intersection is proposed in [4]. The method relies on knowing the speed distribution of cars on each road and uses it to calculate the probability of a vehicle coming to the intersection at the particular ranges of speed that would cause a collision with the ego vehicle. Another probabilistic approach, in this case without relying on historic traffic distributions, is proposed in [5]. The method uses particles to represent the distribution of possible hidden vehicles. Particles are randomly placed along the unseen lanes following a uniform distribution and propagated forward in time with a constant speed. These particles are then used to calculate a risk of collision that can be integrated with a planning algorithm.

The task of planning in intersections with hidden obstacles has also been addressed by formulating the problem as a partially observable Markov decision process (POMDP) [6], [7]. In [6], the authors model hidden vehicles in all currently unseen regions and predict their movements using a Bayesian Model. A similar approach is seen in [7], however here the authors assume that all hidden vehicles arrive in the intersection at the same time as the ego vehicle, given that their required velocity is feasible.

Reachability analysis has been used to over-approximate all possible occupancies of the visible surrounding vehicles, assuming they follow a set of traffic rules [8], [9]. A similar approach has been used to account for possible hidden vehicles by spawning a set of virtual obstacles at the current edge of the unseen region, as in [10], [11]. These methods both assume an orientation of the possible hidden obstacles and that they come from unseen regions that are connected with the left and right border of the lane.

In [9] and [11], the authors validate the computational feasibility of using reachable sets as predictions of other traffic participants. The computation time is reported to be between 9 and 25 ms, which allows for a planning frequency of at least 40 Hz. In [11], the work is extended to also include hidden obstacles, and the approach is implemented on a real vehicle. The authors also validated on a large dataset that the reachable sets always are over-conservative, despite uncertainty in the dynamics and noisy measurements.

One aspect common in all the previously mentioned studies is that they consider unseen regions based only on the current observation from the vehicle’s sensors. This approach might work for unseen areas caused by large static objects, like a building covering an intersection, where more of the unseen region is seen the closer the ego vehicle gets to the intersection. However, when the ego vehicle is close to other dynamic obstacles, the currently unseen regions could shift in position faster than obstacles can possibly move. In these cases, reasoning about which region could be occupied based on previous observations becomes necessary to avoid over-conservative behaviors.

To the best of our knowledge, the only study that considers previous observations to reason about currently unseen regions is [12]. In their implementation, the authors apply their method to track the regions that could be occupied by pedestrians. In contrast, we show the effect of sequential reasoning with possible hidden vehicles and demonstrate the benefits in realistic traffic scenarios where the reachability is computed taking traffic rules into account.

B. Contributions

This work contributes to the autonomous driving field, addressing challenges related to hidden road users by:

1) modeling possible hidden obstacles without assuming prior knowledge about their shape, orientation and, as a result, type of obstacle (e.g. motorcycle or car),
2) proposing an algorithm to reduce the possible states of unseen obstacles by reasoning about previous observations,
3) supporting our proposed model and algorithm with a set of experiments.

Figure 2 demonstrates our algorithm simulated in a real intersection in Fürstenfeldbruck, Germany. The method concludes that possible hidden obstacles can only occupy the dark red regions, even though a much larger region behind the yellow vehicle is currently unseen by the blue AV. The AV can not yet pass through the intersection without intersecting the brighter red regions depicting predictions of possible obstacles. However, in Sec. IV we show how the AV at a future time step utilizes the proposed method to find a trajectory passing closely behind the yellow vehicle.
The motion planning problem for autonomous vehicles is typically expressed as the problem of finding a sequence of future states that respects a set of constraints and either minimizes a cost function or maximizes a reward function [13]. The cost or reward of a plan reflects its performance, e.g. in the sense of comfort and progress towards a given goal. The planning constraints intend to both guarantee a feasible plan (e.g. abiding by the dynamical constraints of the vehicle) and guarantee a safe plan (e.g. not be in collision with any other object) [14]. To guarantee that the ego vehicle is safe and thus not in collision with any obstacle at any future time step, the AV needs to consider possible hidden obstacles that may reach the same area as the AV is planning to enter.

A. Problem Statement

In this work, we study two problems that arise when considering possible hidden obstacles in regions unseen by the AV; modeling hidden obstacles

1) conservatively enough, such that that any possible unseen obstacle is represented and considered, regardless of their size or orientation,
2) not too conservatively, such that obstacles are not considered if they are infeasible given the sequence of past observations and assumed constraints on driving (e.g. maximum speed).

This means finding, (1) an over-approximation of the set of all possible states of hidden obstacles that, (2) contains as few states as possible.

B. Approach

As presented in the related work, previous methods for planning with limited visibility rely on spawning virtual obstacles at the edge of the AV’s field of view and computing the occupancy of the obstacles’ reachable sets. This requires an assumption on which possible obstacle shapes and orientations to consider. In [10] and [11], the authors assume hidden obstacles can only occur where the full width of a lane is unseen. To avoid this assumption, we propose to model possible hidden obstacles with a point mass model. This approach is presented in Sec. III-A.

The assumption that any space currently not seen is occupied, might then be overly conservative, since it may be possible to deduce that no obstacle could have reached a part of the region now outside the AVs field of view. In situations where a region is persistently outside the AVs field of view, e.g. due to obstructing buildings, it may be sufficient to assume that region to be occupied. However, in other environments, regions may not be seen due to moving obstacles obstructing the view, causing rapid changes in which regions are seen and not seen. In Sec. III-B, we reason about which possible hidden obstacles we need to consider and we present an algorithm to compute the set of possible states for all hidden obstacles. Without introducing any new assumptions, or relying on probabilistic guarantees, we can relax the needed planning constraints, increasing the AV’s safe action space.

The integration of possible hidden obstacles in a motion planner is presented in Sec. III-C. An algorithm is presented, exemplifying how possible future occupancies can be predicted and used by the AV to plan a collision-free trajectory.

III. Method

We denote the state of a system at a discrete time $k \in \mathbb{N}$ in the state space $\mathcal{X} \subseteq \mathbb{R}^n$ as $x_k$. The position of a system is given by the projection of its state onto the $xy$-plane, such that

$$\text{proj}_{xy}: \mathcal{X} \to \mathbb{R}^2.$$  (1)

The environment consists of $N$ lanes, $\ell \in \mathcal{L} = \{\ell_1, \ell_2, \ldots, \ell_N\}$. The set $\mathcal{S}^\ell \subset \mathbb{R}^2$ is the set of points in space that belong to a lane $\ell$. Other constraints in a lane $\ell$, e.g. speed limit and driving direction, are given by the set $\mathcal{C}^\ell \subset \mathcal{X}$.

Definition 1 (Valid states in a lane, $\mathcal{X}^\ell$). The set of valid states in a lane, $\mathcal{X}^\ell$, is

$$\mathcal{X}^\ell := \{x \in \mathcal{X} \mid \text{proj}_{xy}(x) \in \mathcal{S}^\ell \land x \in \mathcal{C}^\ell\}.$$  (2)

Note that states outside the set $\mathcal{X}^\ell$ might be physically possible, however, they violate our assumptions. The set of constraints given by $\mathcal{C}^\ell$ do not have to coincide with the driving rules, they can be a relaxed interpretation of them, e.g. it can be reasonable to assume that an unseen vehicle can be traveling 20% faster than the maximum speed of the road. However, this work does not aim to set the level of those constraints, it assumes they have been previously defined.

We denote the set of admissible inputs to a system as $\mathcal{U}$. The input $u_k \in \mathcal{U}$ represents the input applied to a system at time $k$. The discrete time dynamic function $f$ describes state transitions,

$$x_{k+1} = f(x_k, u_k),$$  (3)

with $x_k, x_{k+1} \in \mathcal{X}$ and $u_k \in \mathcal{U}$.

We model points in space to either be free or possibly occupied. The AV has a field of view, i.e. the detected free space in its vicinity, defined as follows.

Definition 2 (Field of view, FoV). We define the FoV$_k \subset \mathbb{R}^2$ as the set of points at the current time step $k$ that are

1) free, i.e. not occupied in space by any obstacle,
2) within the sensor range, i.e. the distance between the point and the sensor position is smaller than a given threshold,
3) in direct line of sight, i.e. along the line from the sensor position to the point in direct line of sight, no point is occupied in space by any obstacle.

At time step $k$, FoV$_k \subset \mathbb{R}^2$, is thus the set of points in space detected as free, i.e. observed as not occupied by any obstacle.
A. Modeling of Possible Hidden Obstacles

Finding the worst-case placement of hidden obstacles outside the field of view is a notoriously difficult problem, similar to the polygon containment problem [15]. Instead of solving this problem, we model possible hidden obstacles as a set of states, $\mathcal{P}$, of a point mass model, filling the region possibly occupied. The benefit of this is that any possible hidden obstacle is considered, regardless of its shape or orientation. This guarantees that no obstacles are missed, however, at the cost of accounting for arbitrarily small obstacles.

Figure 3a gives an example of possible hidden obstacles (in this case motorcycles) in an intersection where the AVs view is limited by another vehicle. All of them are covered in our modeling since our model considers every point in their shape individually. In contrast, [10] and [11] only consider possible hidden obstacles coming from an unseen region that connects the left and right side of a lane.

The set of possible hidden obstacles are modeled individually for each lane, $\mathcal{P}_0^\ell$. For instance, in the example in Fig. 3b, at time $t=0$ we observe three regions possibly occupied by obstacles with states from the sets $\mathcal{P}_0^1$, $\mathcal{P}_0^2$ and $\mathcal{P}_0^3$. Linking the possible hidden obstacles to a lane allows us to know which set of states, $\mathcal{X}^\ell$, is valid for the reachable set of those obstacles. Together with the later introduced Algorithm 1, this linkage can significantly reduce the set of possible hidden obstacles. In Section IV, examples of this will be shown.

Definition 3 (Initial set of states of possible hidden obstacles, $\mathcal{P}_0^\ell$). With no prior observations, the set of possible hidden obstacles in a lane $\ell$ at the initial time step $k=0$ is defined as

$$\mathcal{P}_0^\ell := \{ x \in \mathcal{X}^\ell \mid \text{proj}_{xy}(x) \notin \text{FoV}_0 \}. \tag{4}$$

Figure 3b illustrates the region, in dark red, where possible hidden obstacles can be in each lane, i.e. $\text{proj}_{xy}(\mathcal{P}_0^0)$, $\text{proj}_{xy}(\mathcal{P}_0^1)$ and $\text{proj}_{xy}(\mathcal{P}_0^2)$.

B. Reasoning about Possible Hidden Obstacles

To reason about possible hidden obstacles at our current time step $k$, we use the current field of view, FoV$_k$, and the reachable set of the states of possible hidden obstacles from the previous time step $k-1$.

Definition 4 (One-step reachability, $\mathcal{R}$). Given a set of initial states, $\mathcal{X}^i$, and valid inputs $\mathcal{U}$, we define the one-step reachable set as

$$\mathcal{R}(\mathcal{X}^i) := \{ f(x, u) \in \mathcal{X} \mid x \in \mathcal{X}^i \land u \in \mathcal{U} \}. \tag{5}$$

When considering the constraints $\mathcal{X}^\ell$ of a lane $\ell$, we denote the one-step reachable set as

$$\mathcal{R}^\ell(\mathcal{X}^i) := \mathcal{R}(\mathcal{X}^i) \cap \mathcal{X}^\ell. \tag{6}$$

Definition 5. The states of possible hidden obstacles at the current time step $k$ in a lane $\ell$ are

$$\mathcal{P}_k^\ell = \{ x \in \mathcal{R}^\ell(\mathcal{P}_{k-1}^\ell) \mid \text{proj}_{xy}(x) \notin \text{FoV}_k \}. \tag{7}$$

Lemma. No possible hidden obstacle in a lane $\ell$ at time $k$ can have a state $x \notin \mathcal{P}_k^\ell$.

Proof. From def. 2 of FoV and def. 4 of $\mathcal{R}$ we conclude that

1) the state of a possible hidden obstacle must be in the reachable set of all possible hidden obstacle states from the previous time step, i.e. $x_k \in \mathcal{R}^\ell(\mathcal{P}_{k-1}^\ell)$ and,

2) the projection of a possible hidden obstacle state can not be in our field of view, i.e. $x_k \in \mathcal{X} \mid \text{proj}_{xy}(x_k) \notin \text{FoV}_k$.

therefore the set of all states of possible hidden obstacles at the current time step $k$ must be all states such that (1) and (2) holds, i.e. $\{ x \in \mathcal{R}^\ell(\mathcal{P}_{k-1}^\ell) \mid \text{proj}_{xy}(x) \notin \text{FoV}_k \}$.

Figure 4 illustrates how the set $\mathcal{P}_1^1$ is computed in the example from Fig. 3. At the initial time step $k=0$, $\mathcal{P}_0^1$ is computed as defined in Eq. 4, and its projection is shown in Fig. 4a. The reachable set $\mathcal{R}^1(\mathcal{P}_0^1)$ is highlighted in red in Fig. 4a, and with a dashed line in Fig. 4b. At the next time step $k=1$, the set $\mathcal{P}_1^1$ is computed as in Eq. 7. Its corresponding projection is visualized in dark red in Fig. 4b.

As seen in the figure, the region dashed is concluded to not contain any possible hidden obstacles, even though it is outside FoV$_1$. This is because it is not within the reachable set from the previous time step. Computing the sets $\mathcal{P}_1^2$ and $\mathcal{P}_1^3$, is done correspondingly. In the case of $\mathcal{P}_1^4$, since the
complete lane $\ell_2$ is within FoV, the set $P^\ell_1$ is empty and the reachable set $R^\ell_0(P^\ell_0)$ is thus omitted from the illustration.

Algorithm 1 presents the method used to compute the states of possible hidden obstacles at each time step. It takes as input the current field of view, FoV, and sets for each lane with the states of possible hidden obstacles at the previous time step, denoted $P^\ell_{k-1} := (P^\ell_{k-1}, P^\ell_{k-1}, \ldots, P^\ell_{k-1})$. As output, it provides one set for each lane with the current set of possible hidden obstacles, $P^\ell_k$. We assume that the set of lanes, $\mathcal{L}$, and their corresponding valid states, $\mathcal{X}^\ell$, are available through a map of the environment. The algorithm has two cases: initialization and update, depending on the current time step $k$. Without loss of generality, we assume $k = 0$ at the first run of the algorithm. At this time step, the valid states for the lane are retrieved and the set $P^\ell_0$ is initialized using the available field of view FoV, according to Eq. 4. At subsequent time steps $k > 0$, the previously computed set $P^\ell_{k-1}$ is combined with the current field of view, FoV, to compute $P^\ell_k$ according to Eq. 7.

C. Planning considering Possible Hidden Obstacles

The approach presented in Sec. III-B can be used to compute the set of possible hidden obstacles at the current time step $k$. To use this information for the planning problem discussed in Sec. II-A, a prediction is needed of where those obstacles could reach in the future. This prediction can be computed in numerous ways, as shown in [16], [17].

\begin{algorithm}
\caption{Computing possible hidden obstacle states.}
\begin{algorithmic}[1]
\Input{Current field of view, FoV$_k$, previous set, $P^\ell_{k-1}$.}
\Output{Set of states of possible hidden obstacles, $P^\ell_k$.}
\State $L \leftarrow \text{getLanes}()$
\Foreach{$\ell \in L$}
\If{$k = 0$} \Comment{Initialize}
\State $\mathcal{X}^\ell \leftarrow \text{getAdmissibleStates}(\ell)$
\State $P^\ell_k \leftarrow \{x \in \mathcal{X}^\ell | \text{proj}_x(x) \notin \text{FoV}_k\}$
\Else \Comment{Update}
\State $P^\ell_{k-1} \leftarrow \text{getP}(P^\ell_{k-1}, \ell)$
\State $\mathcal{X}^\ell \leftarrow \mathcal{R}^\ell_k(P^\ell_{k-1})$
\State $P^\ell_k \leftarrow \{x \in \mathcal{X}^\ell | \text{proj}_x(x) \notin \text{FoV}_k\}$
\EndIf
\EndForeach
\State $P^\ell_k \leftarrow (P^\ell_1, P^\ell_2, \ldots, P^\ell_N)$
\State \Return $P^\ell_k$
\end{algorithmic}
\end{algorithm}

In our case, we use reachability analysis to predict regions that may be occupied and thus must be avoided when planning. From Eq. 6, the set of states of possible hidden obstacles at a future time step $i$ in a lane $\ell$ is

$$P^\ell_i = R_\ell(P^\ell_{i-1}),$$  \hspace{1cm} (8)

where the current time step is $k < i$.

Algorithm 2 presents an example of how possible hidden obstacles can be considered when planning. The inputs are the current set of possible states of hidden obstacles in all lanes, $P^\ell_k$, the current ego vehicle state, $x_k$, and a goal region $G \in \mathcal{X}$. The goal region is given by the planned vehicle route and the set $P^\ell_k$ is computed with Alg. 1. The output is a sequence of future states, $\tau = \langle x_k, x_{k+1}, \ldots, x_{k+h} \rangle$, which is guaranteed to be collision free, under the assumption that all other road users stay within the admissible set of states in their lanes, $\mathcal{X}^\ell$.

\begin{algorithm}
\caption{Planning with possible hidden obstacles.}
\begin{algorithmic}[1]
\Input{Current $P^\ell_k$, current ego state, $x_k$, goal region, $G$.}
\Output{Planned trajectory.}
\For{$i \leftarrow (k + 1)$ to $(k + h)$} \Comment{Predict}
\State $P^\ell_i \leftarrow R_\ell(P^\ell_{i-1})$
\EndFor
\State $\tau = \langle x_k, x_{k+1}, \ldots, x_{k+h} \rangle$
\State $T \leftarrow \text{generateTrajectories}(x_k, G)$ \Comment{Plan}
\State $T_s \leftarrow \text{getSafeTrajectories}(T, P^\ell_{k:h})$
\State $\tau \leftarrow \text{getBestTrajectory}(T_s)$
\State \Return $\tau$
\end{algorithmic}
\end{algorithm}
As in Alg. 1, the reachability of each possible hidden obstacle is computed per lane. In line 6 of Alg. 2, the union of all sets of possible states of hidden obstacles at each time step. A set of candidate trajectories, $\mathcal{T}$, is generated, and the set of safe trajectories $\mathcal{T}_s \subset \mathcal{T}$ is computed. These trajectories all guarantee that no part of the ego vehicle at any time step (up to a given horizon $k + h$) is simultaneously in the same position as the projection of $P_{k,k+h}$. The best trajectory $\tau$ is returned (according to a given cost or reward function). If the set $\mathcal{T}_s$ is empty, the trajectory computed at the previous planning instance is returned. If this previous trajectory ended in a state considered safe and was collision checked with over-approximated predictions, it should remain safe under the considered assumptions [18].

IV. EXPERIMENTS

The purpose of our experiments is to demonstrate that motion planning with the proposed modeling and sequential reasoning can reduce the limitations seen in previous works treating hidden obstacles. Namely, we compare the area covered by possible hidden obstacles and we show that in many scenarios, AVs can improve their performance measured in terms of time to pass through an intersection. At the same time, the improved performance comes at no expense of safety. All the scenarios presented here are available together with videos of the simulations at github.com/KTH-RPL-Planiacs/foresee-the-unseen.

A. Baseline method setup

**Baseline method:** We aim to compare our proposed method with the methods presented in [10] and [11]. Since no source code is available from the publications we implemented a modified version of Alg. 1, inspired by these previous works, to be used as a baseline method. The baseline method considers possible hidden obstacles and we show that in many scenarios, AVs can improve their performance measured in terms of time to pass through an intersection. At the same time, the improved performance comes at no expense of safety. All the scenarios presented here are available together with videos of the simulations at github.com/KTH-RPL-Planiacs/foresee-the-unseen.

### Scenarios

The purpose of our experiments is to demonstrate that motion planning with the proposed modeling and sequential reasoning can reduce the limitations seen in previous works treating hidden obstacles. Namely, we compare the area covered by possible hidden obstacles and we show that in many scenarios, AVs can improve their performance measured in terms of time to pass through an intersection. At the same time, the improved performance comes at no expense of safety. All the scenarios presented here are available together with videos of the simulations at github.com/KTH-RPL-Planiacs/foresee-the-unseen.

#### Parameters Used in Experiments

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Sec. IV-B</th>
<th>Sec. IV-C &amp; IV-D</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Planning Frequency</td>
<td>10</td>
<td>10</td>
<td>Hz</td>
</tr>
<tr>
<td>Planning Horizon</td>
<td>5</td>
<td>5</td>
<td>s</td>
</tr>
<tr>
<td>Generated Trajectories</td>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Reference Speed</td>
<td>9</td>
<td>8.33</td>
<td>m/s</td>
</tr>
<tr>
<td>Sensor View Range</td>
<td>50</td>
<td>100</td>
<td>m</td>
</tr>
<tr>
<td>Prediction Horizon</td>
<td>2.3</td>
<td>5.0</td>
<td>s</td>
</tr>
<tr>
<td>Maximum Velocity</td>
<td>12</td>
<td>10</td>
<td>m/s</td>
</tr>
<tr>
<td>Maximum Acceleration</td>
<td>2</td>
<td>3.5</td>
<td>m/s²</td>
</tr>
<tr>
<td>Maximum Deceleration</td>
<td>4</td>
<td>5</td>
<td>m/s²</td>
</tr>
</tbody>
</table>

All these variants are complex situations where the AV needs to consider both visible and possible hidden obstacles.

**Implementation:** All scenarios are visualized and simulated with 0.1s time steps using the CommonRoad toolbox [19]. Collision checking is done with CommonRoad’s Drivability Checker [20] and the integration with the SUMO traffic simulator [21] is done with CommonRoad’s SUMO interface [22]. We implemented Alg. 1 and 2 by creating a simplified version of the approach in [9] using the Python package Shapely [23] to represent the set of states as polygons. To obtain over-approximate predictions and reduce computational complexity [11], we use a single integrator as our dynamical model of the possible hidden obstacles. The input velocity is limited to be between zero and a lane’s maximum velocity, and along the lane’s driving direction.

Throughout the experiments, we use a simple implementation of Alg. 2 to plan trajectories considering possible hidden obstacles. We generate velocity profiles along the centerline of the lane that bring the vehicle to a full stop within the planning horizon. At each time step, we replan and choose the fastest trajectory that at no future time step is in collision with any space reachable by any other object. If no new collision-free trajectory is found at the current time step, the previous trajectory is kept. As in [18], we assume that previous collision-free trajectories are safe since collisions are checked towards conservative predictions of all surrounding objects. The used sets of planning parameters are seen in Tab. I. In Sec. IV-B, the parameters are set to match those in [10], whereas in Sec. IV-C and IV-D the parameters are set to ensure no collisions when interacting with vehicles simulated with SUMO.

**Visualization:** In the following sections, figures visualize the scenarios with four snapshots of the scenes. The snapshots are taken at two different time steps and compare our proposed method with the baseline implementation. The snapshots show the ego vehicle and its plan in blue, dynamic obstacles (e.g. other vehicles) in yellow, static obstacles in gray (e.g. buildings and parked vehicles), regions with possible obstacles in dark red, and predictions of obstacles in a brighter red. The predictions are visualized two seconds into the future, where the possible occupancy at each future time step is represented with a lighter shade of red. Cardinal directions are used when referring to lanes and vehicle routes. The ego vehicle is assumed to always be arriving from the south in the figures.
B. Scenario with Building Limiting the View

In this first experiment, the goal is to demonstrate how reasoning about possible hidden obstacles over time can improve the performance of AVs. We show this by comparing the time taken to complete a scenario both with our proposed method and the baseline method. The scenario and parameters are set to match the ones in [10] in order to compare the baseline method’s results with previous works.

Figure 5.a and Fig. 5.b show the result when planning with the baseline method. In Fig. 5.a, the ego vehicle is planning to stop before the intersection. This is both due to the yellow vehicle coming from the left, but also due to the prediction of possible obstacles coming from the right that would be hidden from the ego vehicle’s sensor view by the gray building. In Fig. 5.b, the ego vehicle is still waiting, since its path forward is blocked by possible hidden obstacles coming from the region behind the yellow vehicle.

Figure 5.c and Fig. 5.d show the same scenario, but this time with Alg. 1 implemented as described in Sec. III. The ego vehicle’s initial behavior when approaching the intersection is the same, however, the area of \( \text{proj}_{xy}(P) \) is greatly reduced behind the yellow vehicle, as can be seen in Fig. 5.c and Fig. 5.d. Observe that the prediction of the possible hidden obstacles behind the yellow vehicle (in Fig. 5.c) travel east or north, but not to the west since a U-turn is not allowed in this road network. This is achieved by using previous sensor views to reason whether something could have reached the now hidden areas, and by updating these possible hidden obstacles per lane. In Fig. 5.d, the ego vehicle is already accelerating and plans a trajectory through the intersection, since it has ruled out the possibility of an obstacle being behind the yellow vehicle.

Figure 6 compares the speed of the ego vehicle when planning with the baseline method and our proposed method. With our proposed method, the ego vehicle can find a path through the intersection after two seconds, without having to come to a full stop. With the baseline method, the ego vehicle needs to come to a full stop and wait. After 4.9 s a plan is found, 2.9 seconds later than with our proposed method. Note that for the baseline method, the results obtained in Fig. 6 closely match the results in [10, Figure 4].

C. Scenarios with Random Traffic

In this section, we show that the potential benefit of this method is not limited to a particular case. We do this by generating 300 scenarios with randomly generated traffic, interacting with each other and the ego vehicle through CommonRoad’s SUMO traffic simulation interface. The scenarios have been populated with randomized vehicles arriving from the east, north, or west. When the scenarios are generated, vehicles are spawned at a sufficient distance from each other, ensuring that SUMO’s controller will prevent them from colliding with each other. To avoid issues with the integration of the CommonRoad road network with SUMO’s traffic simulation, the other vehicles are not performing left turns. Also, a no-stopping zone is added inside the intersection to prevent the ego vehicle from blocking other vehicles. The zone is illustrated in translucent yellow, as seen in Fig. 7.

For each generated scenario, we calculate the difference between the baseline method and our method concerning the time taken to go through the intersection. Figure 8 shows this difference for every scenario, sorted by increasing improvement. The orange bars represent the scenarios where, using our proposed method, the ego vehicle is able to find a gap between the other traffic participants that is not found with the baseline method. This explains the large difference compared to the rest of the scenarios, indicated in blue, where no better gap is found. When no better gap is found, the biggest difference is seen when the last vehicle exits to the east, similar to the example in IV-B. The smallest difference is seen when the ego vehicle is limited by other vehicles, not by possible hidden ones. For instance, when no gap is
found and the last vehicle exits to the west (i.e. the same as the ego vehicle is heading towards), the ego vehicle needs to wait until the last vehicle has passed. In about two out of three scenarios, the improvement is 0.2 s or less, but the performance is never worse than the baseline method. More importantly, no collision occurs in any of the 300 scenarios.

Scenario with Greatest Improvement: Figure 7.a shows the ego vehicle waiting for a gap while using the baseline method. In Fig. 7.b, at $t = 8.3s$, it is not capable to find a safe trajectory. It assumes that some obstacle could be approaching the intersection from the west, due to the occlusion caused by the vehicle traveling south, or from the east, due to the occlusion caused by the vehicle traveling to the east. Figure 7.c also shows the ego vehicle waiting for a gap, but while using our proposed method. The possibility of obstacles coming from the east has already been discarded, and the area where possible hidden obstacles could be behind the vehicle traveling south is smaller. In Fig. 7.d, also at $t = 8.3s$, the ego vehicle is able to find a safe trajectory by considering previous observations. It still considers the possibility of an obstacle coming from the west, but it reasons that it must be farther away from the intersection.

Figure 9 shows the ego vehicle’s velocity both when using the baseline method and when using our proposed method. With the baseline, the ego vehicle needs to wait for 12.9 s extra, until all the other vehicles have passed.

Scenario with no Improvement: In contrast to the scenario presented previously, we now show one of the cases in which sequential reasoning does not lead to any improvement. Figure 10.a shows a large region considered to be possibly occupied by hidden obstacles. In Fig. 10.b, the ego vehicle is still waiting to get into the intersection. Figure 10.c shows that, by reasoning about previous observations, a large region can be concluded to be free from possible hidden obstacles. Unfortunately, in this case, the increase in free area still does not allow the planner to find a path earlier and we observe the same outcome in Fig. 10.d as in Fig. 10.b.

Figure 11 shows the evolution over time of the area that
could be occupied by hidden obstacles in the scenario shown in Fig. 10. When using the baseline method, there are three peaks, corresponding to the three moments in time when another vehicle passes close to the ego vehicle, limiting its sensor view. When this happens, the baseline method assumes a large region in space to suddenly possibly be occupied. In contrast, our proposed method deduces that only a fraction of this region can be occupied.

D. Scenario with Parked Vehicles

In this section, we show that the proposed method not only improves the performance; it can also help to find a safe trajectory in complex scenarios that could otherwise lead to the so-called freezing robot problem, a situation in which the planner considers all possible movements unsafe and the robot gets stuck in its current position [24]. For this reason, we prepare a scenario with parked vehicles, where one is situated close to the no-stopping zone of the intersection. The results obtained in this section are highly dependent on the specific planner used. In our implementation, the ego vehicle is not allowed to enter the no-stop zone of the intersection without a safe plan that reaches the other side. Still, this example demonstrates the importance of reasoning about previous observations to limit freezing situations due to over-conservative estimates about possible hidden obstacles.

Figure 12.a shows the ego vehicle approaching the intersection while using the baseline method. One of the parked vehicles blocks the view of the east lane, causing the ego vehicle to plan a stopping trajectory to avoid possible hidden obstacles entering the intersection. At \( t = 2.0 \text{s} \), in Fig. 12.b the ego vehicle comes to a full stop without being able to plan a safe trajectory that would take it through the intersection. Figure 12.c shows the ego vehicle in the same configuration as in Figure 12.a, but using our proposed method instead. A small difference can be observed in the lane coming to the intersection from the east. Some of the area outside our current field of view is shown free, indicating that based on previous observations and the reachability analysis, no object following the traffic rules could be there (since no reversing is allowed). By \( t = 2.0 \text{s} \), in Fig. 12.d, the ego vehicle has ruled out the possibility of an obstacle coming from the east and is able to find a safe trajectory through the intersection. This is possible thanks to the gap between the two parked vehicles. At each time step between \( t = 0.5 \text{s} \) and \( t = 2.0 \text{s} \), our method reduces the area that could be occupied by hidden obstacles. At \( t = 2.0 \text{s} \), the view of the east lane is...
completely blocked, however, our method can still deduce that no obstacle possibly can be close enough to interfere with a planned path through the intersection.

Figure 13 shows a comparison of the ego vehicle’s velocity when using the baseline method and our proposed method. Using the baseline method, the ego vehicle reaches a full stop and is unable to find a safe trajectory making any progress. With our proposed method, the ego vehicle only needs to decelerate momentarily before being able to discard the option of incoming traffic from the east, thus avoiding the freezing robot problem.

Fig. 13. With our method the ego vehicle finds a trajectory through the intersection after 1.9 seconds, whereas the baseline method comes to a stop and is unable to continue.

V. CONCLUSIONS AND FUTURE WORK

This work focuses on the problem of safe motion planning in presence of possible hidden obstacles. Specifically, this work aims to (1) model hidden obstacles such that any possible unseen traffic participant is considered and (2) use previous observations to remove infeasible obstacle states. By modeling hidden obstacles as a set of states of a point mass model, every possible unseen traffic participant is guaranteed to be considered, regardless of their shape or orientation, and, as a result, type of obstacle. Using sequential reasoning, based on our previous observations and reachability analysis, our experiments show that performance can be improved at no expense of safety.

Previous work has successfully validated the use of reachability to compute over-approximate possible occupancies of other road users in real-time, as described in I-A. Hence, a natural next step for this work is to analyze the real-time behavior of our proposed algorithm. Other possible extensions include using state estimates of tracked vehicles, e.g. available from an object tracker, to shrink the regions predicted to be occupied by hidden obstacles. To further reduce the possible states of hidden obstacles, more assumptions could be included, both through dynamical constraints and through a richer set of traffic rules. Additionally, the analysis of the method could be extended to more scenarios with different road networks.

REFERENCES


