



HAL
open science

Safe Geometric Speed Planning Approach for Autonomous Driving through Occluded Intersections

Renaud Poncelet, Anne Verroust-Blondet, Fawzi Nashashibi

► **To cite this version:**

Renaud Poncelet, Anne Verroust-Blondet, Fawzi Nashashibi. Safe Geometric Speed Planning Approach for Autonomous Driving through Occluded Intersections. ICARCV 2020 - 16th International Conference on Control, Automation, Robotics and Vision, Dec 2020, Shenzhen, China. hal-02967740v2

HAL Id: hal-02967740

<https://hal.science/hal-02967740v2>

Submitted on 19 Oct 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Safe Geometric Speed Planning Approach for Autonomous Driving through Occluded Intersections

Renaud Poncelet¹, Anne Verroust-Blondet¹ and Fawzi Nashashibi¹ *

Abstract

Autonomous driving in urban environment needs to anticipate a number of dangerous events, such as the presence of moving vehicles in occluded areas. This paper presents an approach computing a safe motion along a fixed path in an urban environment with dynamic vehicles that may be occluded. The method works on the time-path space and uses a visibility graph to compute the speed profile, considering both safety and comfort. Evaluations performed on CARLA simulator on several typical scenarios show that the approach is able to drive safely in presence of hidden obstacles.

1 Introduction

Autonomous driving in urban areas requires behaving safely in the presence of other vehicles that can be visible or occluded by physical obstacles. If we observe human drivers, we notice that they slow down when approaching an occluded area and drive cautiously until they realize that the path is clear. In fact, they assume that at any moment a vehicle may emerge from the occluded area.

Driving in a well-defined urban road infrastructure links the vehicle trajectory strictly enough to follow the bounds which are indicated by lane markings, once the maneuver has been determined. Thus, in what follows, we assume that the vehicle's trajectory is given and we use the path-velocity decomposition introduced by Kant and Zucker [1] to focus on a speed profile adaptation method that ensures the safety of the motion planning.

In what follows, we present a geometric method computing a safe speed profile for an autonomous vehicle navigating in the presence of dynamic obstacles that may be occluded. Our approach uses a visibility graph path planning method to compute a safe speed profile in the Time-Path domain in which the potential risks from occlusion and limited sensing are considered. Moreover, we take into account passenger comfort by constraining the speed and the acceleration of the vehicle.

2 Related Research

The problem of occluded areas has been addressed in numerous recent approaches [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]. They can be classified in different families, according to the way they represent these regions:

2.0.1 Particle filters

In order to take into consideration risks from occluded areas and the limited range of sensors, some approaches use particle filters. In [9], particles are generated randomly in occluded areas with random speed. A decision to cross is made considering particle kinematics. In [12], the particles are also spread to represent their possible deviation from the center of the lane. Here the particles are used to compute a risk cost which combined with a speed cost allows a speed along a given path to be computed. Particle filters approaches give a probabilist representation of the environment but do not have worst case assumption.

2.0.2 Virtual vehicles

Other approaches model the worst case scenario by using virtual vehicles to represent all potential vehicles in occluded areas [3, 5, 6, 7, 11, 8]. For a risk evaluation purpose, Damerow et al. [3] position one virtual car for each relevant lane at the occluded position closest to the upcoming intersection. Similarly, in Orzechowski et al. [5], virtual vehicles are placed at visibility limits. A fail-safe maneuver is then computed, following the reachable set approach introduced in [13]. In [6], virtual vehicles are used to find rules for a safe motion or a safe stop for free drive, follow drive, right-of-way maneuvers and give-way maneuvers. They consider uncertainties and the worst case scenario to solve an optimization-based planner. In [7], virtual vehicles are placed at visibility limits on roads that are likely to cross the ego-vehicle's path. They are modeled with an infinite length and are considered as driving at maximum speed. In [11], probability of occurrence is computed using traffic density to position virtual vehicles in occluded areas. Virtual vehicles are supposed to arrive at the same time as the ego-vehicle at the junction as long as they keep a realistic speed.

2.0.3 Other models

Hoermann et al. [2] introduce an unobservable regions layer in their grid-based representation of the environment to estimate whether a region is safe to drive in at maximum speed. To be safe but not to overcautious Naumann et al. [10] focus on conflict zones inside junctions and compute the speed by minimizing a cost. They consider the trajectories of potentially occluded objects along with the probability of their occurrence to estimate the cost caused by them.

Numerous approaches solve the problem as a Partially Observable Markov Decision Process (POMDP). In [4], possible actions are relative to acceleration: for each user, optimal belief action utility is computed, a common belief is computed for all invisible users and then the minimum

*¹Inria Paris, 2 rue Simone Iff 75012 Paris FRANCE
{renaud.poncelet,anne.verroust,fawzi.nashashibi}@inria.fr

belief action utility over users is computed with the sum of all actors belief action utilities. In [7], an observation model is defined depending on the field of view and virtual vehicles have a specific transition model. Lin et al. [8] use a belief tree search built from possible ego-vehicle actions: speed-up, keep speed or brake. Virtual vehicles are set at visibility limits of the ego-vehicle and they are considered as real vehicles. Positions, speeds, and accelerations of other vehicles are supposed to be known and all possible future paths are considered.

In our work, we introduce virtual vehicles to represent occluded parts of the road. We also take into account comfort and safety [14] as acceleration is bounded and a safe state is planned before intersections.

3 Approach

In what follows we assume that a map of the environment is given with all the information on the road topology. A path s is computed off-line given the initial position and a global destination of the ego vehicle. It is assumed to be collision-free regarding static obstacles. We assume that the positions and the velocities of the other vehicles are known for a short period of time once they are visible from the ego vehicle. However, when they are close to an intersection, we assume that they can either turn right, turn left or go straight when possible. Our goal here is to adapt the speed of the ego vehicle to obtain a safe and comfortable trajectory when it follows its path in the presence of dynamic obstacles and considering occluded areas. This is done every time step in the Time-Path domain.

Before presenting the whole model, we first explain our approach in the case where there are no occluded areas. Two processes are executed in parallel, with different time steps and different frame rates:

1) for each time step t_i , our algorithm computes the longitudinal motion $s(t)$ for the ego vehicle, considering the interacting vehicles, i.e. the list \mathcal{N} of surrounding vehicles that intersect the part of the path of the ego vehicle limited to $[s(t_i), s_h]$ during the period $\Delta T = [t_i, t_i + t_h]$, where t_h is a time horizon and s_h the visibility horizon.

2) for each time step τ_j , the longitudinal motion is simulated through a car model (Section 3.3). The time interval between two consecutive time steps τ_j and τ_{j+1} is significantly smaller than that of two consecutive time steps t_i and t_{i+1} , as illustrated in Figure 2.

In the rest of the paper, s will denote the path, v the velocity and a the longitudinal acceleration of the ego vehicle.

3.1 Computing a speed profile in the Time-Path domain

We follow [1] and compute a longitudinal motion $s(t)$ for the ego vehicle during the period $\Delta T = [t_i, t_i + t_h]$ along the path s and work in the Time-Path domain $TP = (t, s)$. The longitudinal motion $s(t)$ has to be a monotonically increasing function of t with constraints on the slope and on the curvature of $s(t)$ deduced from the constraints on the velocity $v(t)$ and the acceleration $a(t)$ of the ego vehicle. Moreover, in order to be a safe motion, $s(t)$ can be deduced from the visibility graph built in TP as in [1], considering

the set \mathcal{N} of vehicles that may intersect s during ΔT . The computation of $s(t)$, $v(t)$ and $a(t)$, $t \in \Delta T$ is carried out in three steps:

1. A polygonal zone, called TP obstacle, is inserted in TP for each vehicle belonging to \mathcal{N} (Section 3.1.1)
2. A polygonal line $\tilde{s}(t)$, candidate for $s(t)$, is built in TP (this is detailed in Section 3.1.2).
3. $\tilde{s}(t)$ is smoothed in order to respect the constraint on the longitudinal acceleration (described in Section 3.1.3).

Steps 2 and 3 are illustrated in Figure 3.

3.1.1 TP obstacles

We assume that the vehicles belonging to \mathcal{N} have a constant velocity during the time period ΔT . Then, for each of them, a TP obstacle is built: it corresponds to the polygonal zone occupied in TP by the vehicle during the period ΔT . To ensure safety, comfort and to take into account possible changes in velocity during the time period ΔT , the occupied zone in TP by the vehicle is expanded in time of a time gap t_{gap} and in path of a *margin*. In order to take into account maneuver uncertainties, we consider that the ego vehicle cannot know which maneuver another vehicle will choose when it enters a road junction before being fully engaged in one lane. Thus, for the scenario in Figure 1 in which the ego-vehicle will cross an intersection where another vehicle is coming from the right, paths a, b and c are considered and the region that may be occupied by the vehicle during ΔT is represented by the union of the TP obstacles corresponding to maneuvers a, b and c.

3.1.2 Visibility graph method

Once the TP obstacles have been built, a candidate path $\tilde{s}(t)$ for $s(t)$ is computed on TP . As mentioned above, it is a path composed of line segments having a positive slope bounded by the maximum speed value v_{Max} , joining point $(0, 0)$ and a point belonging to the vertical line corresponding to time $t_i + t_h$ or belonging to the horizontal line corresponding to s_h . This path connects a subset of the vertices of the TP obstacles and it is built by a visibility graph-based method ([1]).

3.1.3 Achieving acceleration constraints

The candidate path $\tilde{s}(t)$ for $s(t)$ built previously respects the speed limits but needs to be smoothed as it may not satisfy the acceleration bounds. This is done by smoothing the velocity $\tilde{v}(t)$ with a saturation on the acceleration $\tilde{a}(t)$ of the candidate path $\tilde{s}(t)$, with $\tilde{v}(t_i) = v_i$:

$$\tilde{a}(t_i) = \left(\frac{\tilde{s}(t_i + dt) - \tilde{s}(t_i)}{dt} - v_i \right) / dt \quad (1)$$

and,

when $t_i < t < t_i + t_h$:

$$\tilde{v}(t) = \frac{\tilde{s}(t + dt) - \tilde{s}(t)}{dt}, \quad (2)$$

$$\tilde{a}(t) = \frac{\tilde{v}(t + dt) - \tilde{v}(t)}{dt} \quad (3)$$

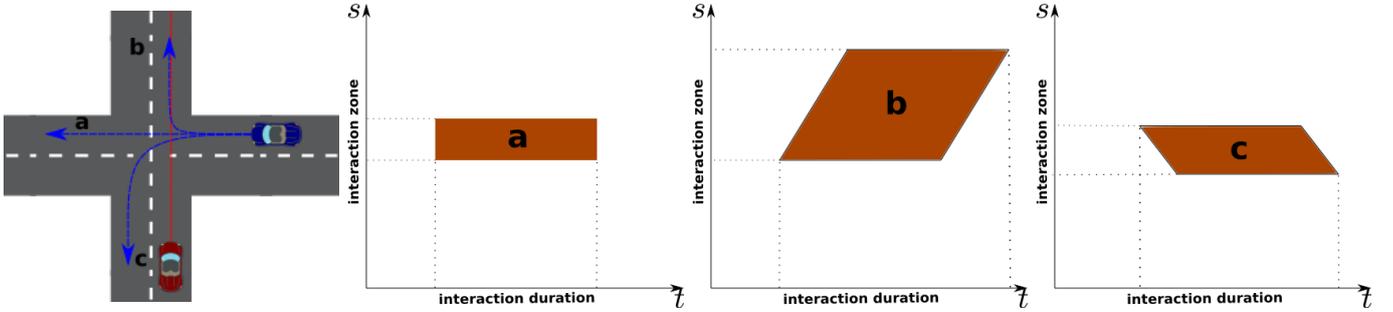


Figure 1: The three possible maneuvers for the blue vehicle coming from the right and their corresponding TP obstacles in TP .

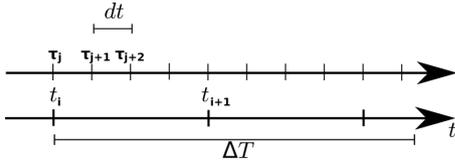


Figure 2: The different time steps used in our approach. A longitudinal motion is planned every time step t_i considering the period ΔT and the control of the ego vehicle is computed every τ_j

Then, the smoothed path $s(t)$ is defined by :
 $s(t_i) = s_0$, $v(t_i) = v_i$ and,
when $t_i \leq t < t_i + t_h$:

$$a(t) = \min(\max(\tilde{a}(t), a_{Min}), a_{Max}), \quad (4)$$

$$v(t) = \min(\max(\tilde{v}(t) + a(t) \times dt, 0), v_{Max}), \quad (5)$$

$$s(t + dt) = s(t) + dt \times v(t) \quad (6)$$

Once $s(t)$ is computed, we have to check whether it intersects a TP obstacle, as in the example of Figure 3. In this case, a new candidate $\tilde{s}(t)$ is built and new smoothed path $s(t)$ is computed: $\tilde{s}(t)$ corresponds to a polygonal line of the visibility graph that passes on the right of the intersected polygon.

3.2 Safe state before intersections

Before validating $\tilde{s}(t)$ as a safe plan, we ensure that the plan does not end in an intersection to remove uncertainties about safety when crossing the intersection. We decide to let the ego vehicle drive as fast as possible to the junction while preserving a safe distance to let it brake before the intersection.

In order to model this behavior in the TP domain, we reduce the visibility limit for the ego vehicle considering that the visibility horizon s_h is moved just before the intersection. Moreover, to enforce the vehicle to stop when reaching the visibility limit, we add a stopgap at in the graph at :

$$s_{brake} = s_h - d_{brake} \quad (7)$$

where d_{brake} is the distance required to brake when the vehicle's speed is equal to v_{Max} : if $\tilde{s}(t) > s_{brake}$ then $\tilde{s}(t) = s_{brake}$ as in the Fig. 3.

3.3 Car simulation

In parallel, at each time step τ_j a longitudinal controller adapts the jerk of the vehicle. It uses as input $\hat{s}(\tau_j)$, $\hat{v}(\tau_j)$,

the measured values in the simulation of $s(t)$ and $v(t)$ at $t = \tau_j$ and $s(\tau_j)$, $v(\tau_j)$, Then it computes $\hat{a}(\tau_{j+1})$, $\hat{v}(\tau_{j+1})$ and $\hat{s}(\tau_{j+1})$ for the car simulator. The initial speed considered at time t_{i+1} to compute the longitudinal motion for the next period in Section 3.1.3 is $v_{i+1} = \hat{v}(t_{i+1})$.

3.4 Tackling occlusions and limited sensor range

Let us now extend our approach to take into account the existence of occluded areas on the road. To follow what a human driver may do in this situation we introduce virtual vehicles and a virtual horizon line (Section 3.4.1) and adapt the longitudinal motion $s(t)$ to be able to react safely without being blocked (Section 3.4.2).

3.4.1 Virtual vehicles and virtual horizon

To represent all the potential risks from occluded areas belonging to the lanes crossing the path of the ego vehicle, we choose to introduce virtual vehicles on the visibility borders, as in [3, 2, 5, 10, 6]. In order to plan for the worst case, we suppose that the virtual vehicle is of infinite length and that it drives at the maximum speed possible, except when there exists a vehicle moving in the same lane in front of the visibility boundary. In this case, we assume that the virtual vehicle has the same speed as this vehicle. Moreover, when the visibility boundary is close to a road intersection, we assume that it may follow any path, either turning left, turning right or going straight when it is possible. Thus we create the following TP obstacles:

At time step t_i , if a border of an occluded area belongs to a lane crossing the path of the ego vehicle during ΔT , and if this border is very close to the road junction, a TP obstacle is inserted in TP to represent the region occupied by a virtual vehicle positioned at the visibility boundary and driving at the maximum speed during ΔT or at the same speed as the previous vehicle when such a vehicle exists. As a virtual vehicle is supposed be of infinite length, the corresponding polygon is bounded by the right boundary of TS , i.e. the vertical line $t = t_i + \Delta T$ (Figure 4).

Areas outside the sensor range may also contain hidden obstacles. Thus, if we want to ensure a safe behavior, the ego vehicle must stop when reaching the visibility horizon value s_h in its path and the horizontal line $s = s_h$ will be considered as an obstacle when computing the candidate path $\tilde{s}(t)$.

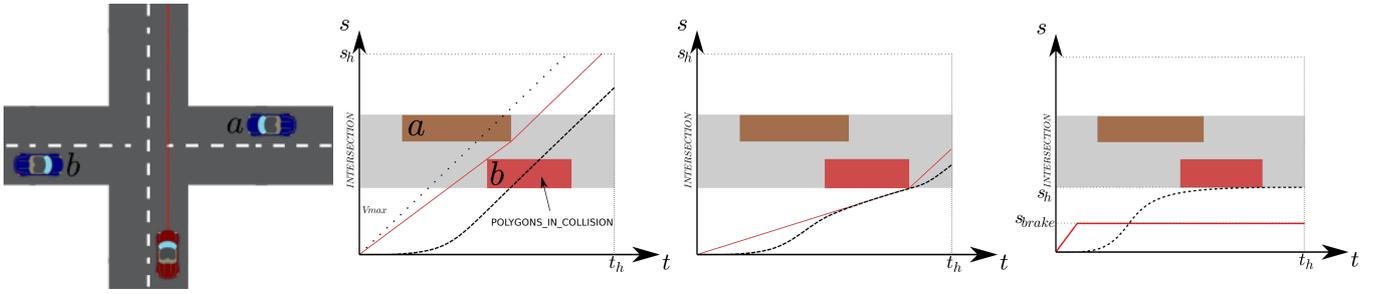


Figure 3: Computation of a trajectory curve. On the extreme left, an overview of the scene. On the middle left, the candidate path \tilde{s} (red line) from the visibility graph method as in [1] and the smoothed version (black dashed line) of \tilde{s} intersects one TP obstacle (in red). At $t = 0$ the ego-vehicle is stationary and its speed is equal to 0. On the middle right, a new candidate path \tilde{s} is built, which passes after the polygon in collision. On the extreme right: the visual horizon (s_h) is virtually reduced before the intersection area (in grey). A new candidate path \tilde{s} is built considering a safe state before the intersection thanks to the constrain $\tilde{s} < s_{brake}$. The last smoothed version s is safe. We notice that a solution can always be found under two conditions : The initial situation have to be safe i.e the ego-vehicle can brake without entering an intersection, and obstacles can't collide the ego-vehicle when it is stopped.

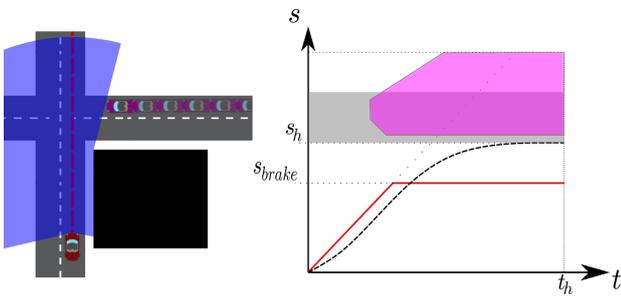


Figure 4: A scenario with an occluded area. Left: the ego vehicle follows a path (red dashed line) and an obstacle occludes a part of the road (the visible area corresponds to the transparent blue area) where a virtual vehicle has been created (drawn in purple). Right: the corresponding TP obstacle (note that its shape is the addition of the three possible cases seen in Figure 1), the candidate path $\tilde{s}(t)$ and the safe path $s(t)$ for the ego vehicle in TP .

3.4.2 Speed planning computation

Once the TP obstacles corresponding to the virtual and the visible vehicles and the visibility horizon line have been inserted in TP , we have to compute a candidate path $\tilde{s}(t)$ for the ego vehicle. We have two cases to consider:

- If the limits of the occluded areas are sufficiently far away from the intersection with the path of the ego vehicle, the corresponding TP obstacles will be small. In this case the ego vehicle will have enough visibility to be able to cross the road junction safely and a polygonal line $\tilde{s}(t)$ built as previously will pass through the left part of the zone corresponding to the road junction in TP (as in case b of Figure 5).
- When the visibility is not sufficient, our solution should be safe but not overcautious. The visibility horizon s_h is moved to the smallest value of s of the first TP obstacle associated to a virtual vehicle to let it brake if another vehicle comes from the occluded area.

Then a corresponding smoothed version $s(t)$ of $\tilde{s}(t)$ is computed as in Section 3.1.3.

4 Experiments

Experimental evaluations have been performed using CARLA simulator [15] on an environment consisting of several road junctions, spaced 40 m apart, with a building hiding some parts of the road. (Video example at : <https://team.inria.fr/rits/membres/renaud-poncelet/>) All the other vehicles have been controlled by a CARLA agent. They follow a random path guided by the lanes and they drive at maximum speed v_{Max} except when another vehicle is in a control area of ten meters, in which case they brake.

Several scenarios have been evaluated where the ego vehicle has to cross a road junction with three different behaviors :

- *Omniscient*: the ego vehicle knows the position and the speed of all the vehicles present in the scene, including the occluded ones.
- *Naive*: the ego vehicle reacts with respect to the visible vehicles and does not take into account the occluded parts of the scene.
- *SGSPA for Safe Geometric Speed Planning Approach*: This is the complete approach presented in this paper.

For each behavior the motion of the ego-vehicle is simulated using a longitudinal control described in Section 3.3. The following parameters have been used during the tests: $v_{Max} = 8.3 \text{ m/s}$, $a_{Max} = 2.5 \text{ m/s}^2$, $a_{Min} = -4 \text{ m/s}^2$, $\Delta T = 5 \text{ s}$, $t_{i+1} - t_i = 1/5 \text{ s}$, $dt = 1/20 \text{ s}$, $t_{gap} = 2 \text{ s}$, $margin = 1 \text{ m}$. The value of s_h is given by the equation: $s_h = 2v_{Max} \times t_h$.

4.1 Crossing an X junction

The first situation is an X junction where the road leading from the right is partially occluded by a building. The ego vehicle has to cross the intersection and go straight. Two scenarios are presented:

1. *Empty occluded area*. Let us first consider the motion computed by *SGSPA* illustrated in Figure 5:
 - (a) When the ego vehicle is far from the intersection ($t = 0.0 \text{ s}$), the border of the occluded area is

near the intersection and the *TP* obstacle associated to the virtual vehicle is very long, thus the algorithm plans to go at maximum speed and to brake when the braking distance is reached.

(b) Later ($t = 3.6$ s), the ego vehicle reaches the intersection and has enough visibility to cross it safely as the border of the occluded area is far away.

(c) Finally ($t = 5.8$ s), the ego vehicle is crossing and its visibility is complete.

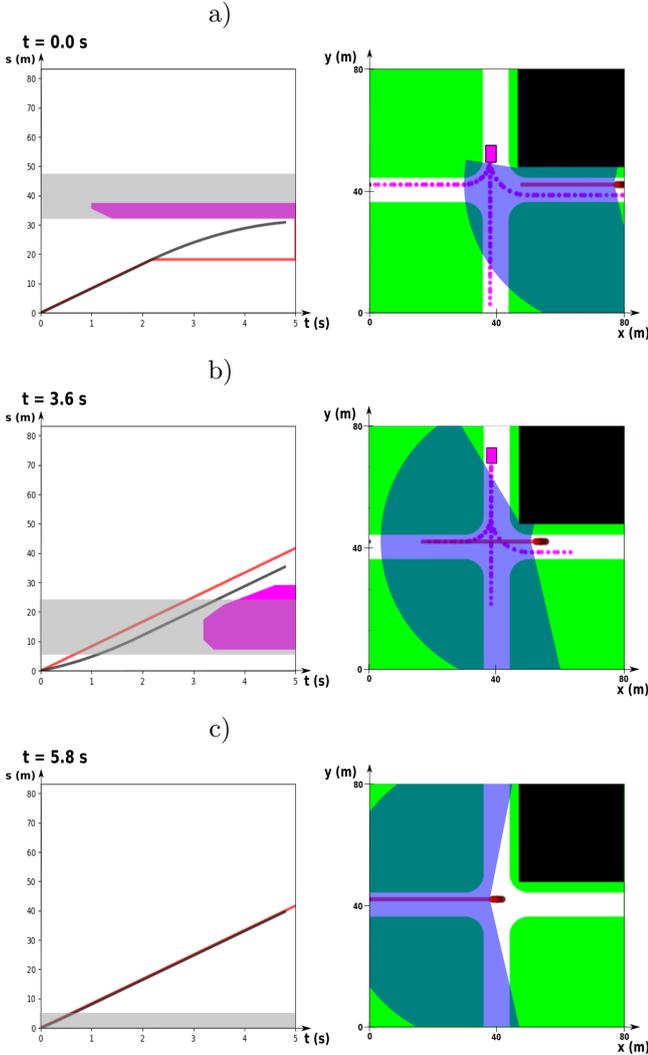


Figure 5: Empty occluded area. Different time steps of SGSPA. Virtual vehicles: possible paths and *TP* obstacles are drawn in pink. Ego vehicle: $\tilde{s}(t)$ is drawn in red and $s(t)$ in black.

As expected, in Figure 6, we can notice that the *Omniscient* and the *Naive* approaches both plan their speed in the same way as there is no hidden vehicle.

2. *One car is on the occluded part of the road.* In this scenario, the other vehicle drives at v_{Max} , reaches the junction approximately at the same time step as the ego-vehicle and goes straight onto the junction. The ego vehicle has to give way to the other car.

Let us go through each approach portion by portion in Figure 7:

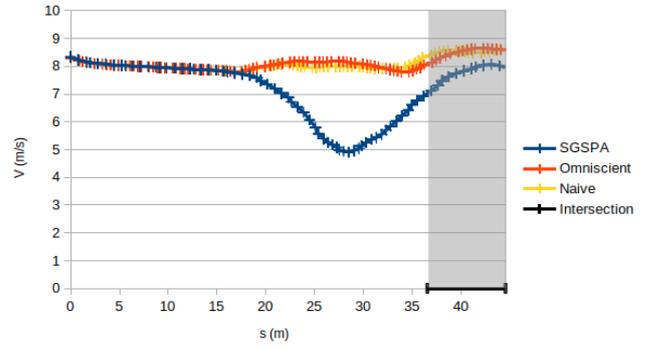


Figure 6: Empty occluded area : measured speed over curvilinear abscissa for the three approaches.

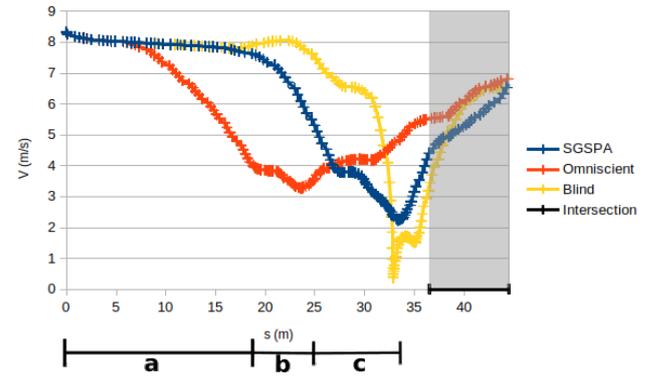


Figure 7: Measured speed over curvilinear abscissa for the three methods when a vehicle is occluded

- In portion a, where the ego vehicle is far away from the junction and cannot see the hidden vehicle, *SGSPA* and *Naive* approaches drive at maximum speed and the *Omniscient* approach slows down. In fact, *SGSPA* reacts like *Naive* to increase its visibility and *Omniscient* takes into account the hidden vehicle.
- In portion b, *SGSPA* begins to brake to be able to stop before the junction while *Naive* goes still very fast.
- In portion c, *Naive* brakes hard to avoid a collision with the other vehicle, which is now visible, and *Omniscient* begins to speed up because its speed is low enough to let the other vehicle pass.

All passed safely the intersection, but *Omniscient* and *SGSPA* approaches computed comfortable motions whereas it was not the case for *Naive*.

4.2 T junction

Let us now compare the three methods on a scenario inspired by [6]: the ego vehicle has to turn left at a T junction where the visibility is limited. As illustrated in Figure 8, both sides are occluded: on the left hand by a building and on the right hand by a line of stopped cars on the road. The visibility is more limited on the right hand side where the arrival of a hidden vehicle enforces the ego vehicle to give way to it. In this example, the *Naive* approach can't brake hard enough to avoid the collision while the

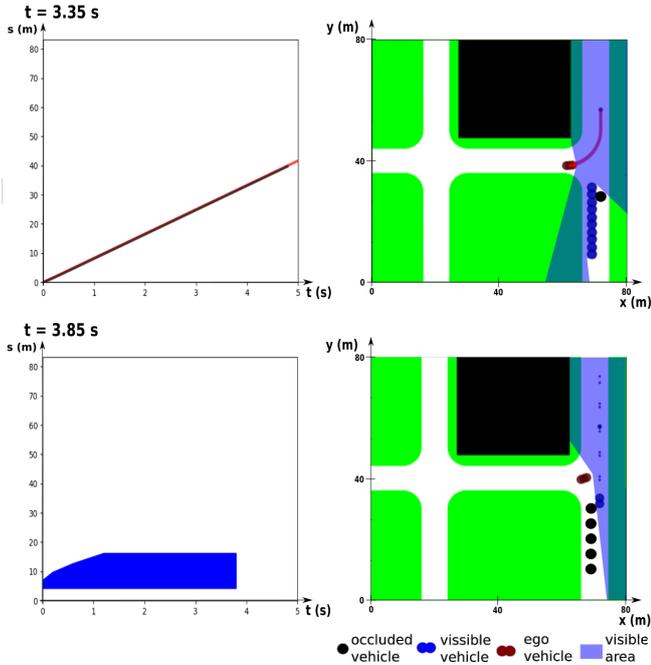


Figure 8: T junction with limited visibility with no consideration of the occluded parts of the scene. *Naive* approach does not anticipate the arrival of the hidden vehicle on the right and the collision cannot be avoided.

other approaches reduce their speeds to let the other vehicle pass, as shown in Figure 9.

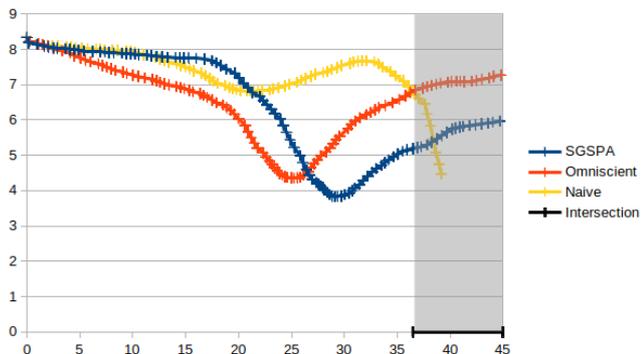


Figure 9: T junction: measured speed over curvilinear abscissa for the three methods.

4.3 X junction with ten cars

Finally, to evaluate the safety of the approach in more complex cases, we created 50 scenarios for the X junction environment of Section 4.1 with ten vehicles randomly positioned on the roads and following a random path. Ten cars may not all interact with the ego-vehicle at the same time : for example, in Fig. 10, only two cars are involved. Note that not all the vehicles are hidden by the building. In all the scenarios, the ego vehicle had to cross the X junction but with two different behaviors :

- *SGSPA*, where the ego vehicle is enforced to stop before the junction if it cannot pass through it.
- *GSPA for Geometric Speed Planning Approach without ensuring a safe state before intersections.*

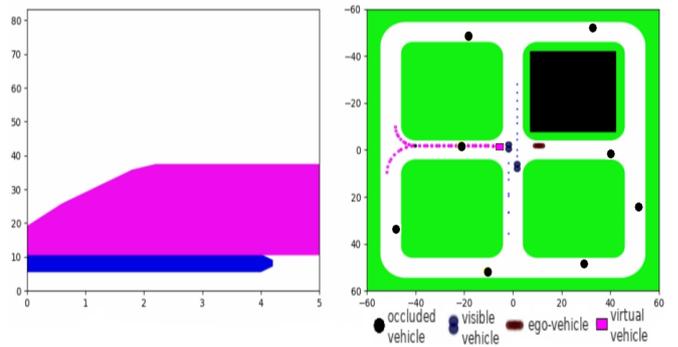


Figure 10: X Junction with two other cars arriving at the intersection at the same time . The ego vehicle is stationary before the intersection. The blue TP corresponds to the car arriving on the left side of the ego vehicle. The TP corresponding to the real car coming from the right is masked by the virtual TP that represents the possibility of the presence of a car hidden by it on the front track.

Let us compare the success rate and the time spend in the junction in both case and discuss the cases where a crash had occurred:

- For *SGSPA*, we obtained 100 % success and the mean time spend by the ego vehicle in the junction was 3.8 s .

- For *GSPA without safe state before intersections*, nine collision occurred (success rate: 82 %), among which for eight of them the ego vehicle was at rest in the middle of the junction. In fact, it is not really a passive safety [14], as the vehicle was inside the junction. In the last case the ego-vehicle was moving but the other vehicle in collision was behind the ego-vehicle. The mean time spent inside the junction in case of success was 4.3 s .

5 Conclusion

In this paper we proposed a geometric approach computing a safe and comfortable speed planning for autonomous driving in urban area with occlusions. Evaluations show that *SGSPA* achieves both safety and comfort in all scenarios. The approach is cautious but not over defensive, thanks to the speed profile we choose when approaching an occluded region: we decide to move quickly to the intersection in order to increase the ego vehicle visibility while keeping enough space to brake.

This approach can be generalized to other dynamic obstacles, such as pedestrian, considering the possible existence and intent of hidden agents as Nager et al. [16].

However, in some really limited visibility cases, we cannot ensure that the ego-vehicle will not be blocked in a “freezing state” as discussed in [8]. In this case, a human driver will drive very slowly in order to increase its visibility. Work in progress extends our approach to mimic such a behavior, knowing that it is not completely satisfying regarding safety and comfort.

References

- [1] K. Kant and S. W. Zucker, “Toward Efficient Trajectory Planning: The Path-Velocity Decomposition,” *The International Journal of Robotics Research*, vol. 5, no. 3, pp. 72–89, 1986.

- [2] S. Hoermann, F. Kunz, D. Nuss, S. Renter, and K. Dietmayer, “Entering crossroads with blind corners. A safe strategy for autonomous vehicles,” in *IEEE Intelligent Vehicles Symposium (IV)*, pp. 727–732, June 2017.
- [3] F. Damerow, T. Puphal, Y. Li, and J. Eggert, “Risk-based driver assistance for approaching intersections of limited visibility,” in *IEEE International Conference on Vehicular Electronics and Safety, (ICVES)*, pp. 178–184, IEEE, June 2017.
- [4] M. Bouton, A. Nakhaei, K. Fujimura, and M. J. Kochenderfer, “Scalable decision making with sensor occlusions for autonomous driving,” in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2076–2081, May 2018.
- [5] P. F. Orzechowski, A. Meyer, and M. Lauer, “Tackling Occlusions Limited Sensor Range with Set-based Safety Verification,” in *IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 1729–1736, Nov. 2018.
- [6] O. S. Tas and C. Stiller, “Limited Visibility and Uncertainty Aware Motion Planning for Automated Driving,” in *IEEE Intelligent Vehicles Symposium (IV)*, pp. 1171–1178, June 2018.
- [7] C. Hubmann, N. Quetschlich, J. Schulz, J. Bernhard, D. Althoff, and C. Stiller, “A POMDP Maneuver Planner For Occlusions in Urban Scenarios,” in *IEEE Intelligent Vehicles Symposium (IV)*, pp. 1909–16, June 2019.
- [8] X. Lin, J. Zhang, J. Shang, Y. Wang, H. Yu, and X. Zhang, “Decision making through occluded intersections for autonomous driving,” in *IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 2449–2455, Oct. 2019.
- [9] P. Narksri, E. Takeuchi, Y. Ninomiya, and K. Takeda, “Crossing Blind Intersections from a Full Stop Using Estimated Visibility of Approaching Vehicles,” in *IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 2427–2434, Oct. 2019.
- [10] M. Naumann, H. Königshof, M. Lauer, and C. Stiller, “Safe but not Overcautious Motion Planning under Occlusions and Limited Sensor Range,” in *IEEE Intelligent Vehicles Symposium (IV)*, pp. 140–145, June 2019.
- [11] P. Schorner, L. Tottel, J. Doll, and J. M. Zollner, “Predictive trajectory planning in situations with hidden road users using partially observable markov decision processes,” in *IEEE Intelligent Vehicles Symposium (IV)*, pp. 2299–2306, June 2019.
- [12] M. Y. Yu, R. Vasudevan, and M. Johnson-Roberson, “Occlusion-aware risk assessment for autonomous driving in urban environments,” *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 2235–2241, 2019.
- [13] M. Althoff and S. Magdici, “Set-based prediction of traffic participants on arbitrary road networks,” *IEEE Trans. Intelligent Vehicles*, vol. 1, no. 2, pp. 187–202, 2016.
- [14] S. Shalev-Shwartz, S. Shammah, and A. Shashua, “On a formal model of safe and scalable self-driving cars,” *CoRR*, vol. abs/1708.06374, 2017.
- [15] A. Dosovitskiy, G. Ros, F. Codevilla, A. López, and V. Koltun, “CARLA: an open urban driving simulator,” in *1st Annual Conference on Robot Learning, CoRL 2017, Mountain View, California, USA, November 13-15, 2017, Proceedings*, vol. 78 of *Proceedings of Machine Learning Research*, pp. 1–16, PMLR, 2017.
- [16] Y. Nager, A. Censi, and E. Frazzoli, “What lies in the shadows? Safe and computation-aware motion planning for autonomous vehicles using intent-aware dynamic shadow regions,” in *International Conference on Robotics and Automation (ICRA)*, pp. 5800–5806, May 2019.