2015 European Control Conference (ECC)
July 15-17, 2015. Linz, Austria

A Path Planner for Autonomous Driving on Highways Using a Human Mimicry Approach with Binary Decision Diagrams

Laurène Claussmann*, Ashwin Carvalho†, Georg Schildbach†

Abstract—**This paper considers the problem of path planning for autonomous ground vehicles on highways with regular traffic. The goal is to select a desired trajectory from a set of parameterized candidate trajectories such that some criterion is optimized. This selection is subject to avoiding collisions, respecting the traffic rules, and eliciting smooth behavior for passenger comfort. The desired trajectory is fed as a state reference to a path follower which uses Model Predictive Control (MPC) to compute the steering and braking actions. This work proposes a human mimicry approach for the path planning problem by considering a Binary Decision Diagram (BDD). The binary decision diagram applies a tree search algorithm coupled with a truth table decision process. Simulation and experimental results are presented to verify the real-time feasibility and demonstrate the effectiveness of the proposed algorithm.**

I. Introduction

A driver, and soon an autonomous ground vehicle, is confronted with many different situations, such as parking, city driving, or highway driving. All these situations are very specific in the sense that they demand different vehicle responses, and hence different control algorithms. The long desired planning horizon, the nonlinearity of the vehicle dynamics, and the complexity of a dynamic traffic environment result in a challenging combinatorial, nonlinear, optimal control problem. As part of a solution for autonomous ground vehicle, most existing control algorithms rely on a path planner as a higher-level logic [1].

Path planning in dynamic environments is a difficult problem because of a large number of candidate solutions [2, 3]. One widespread approach is to grid the overseen terrain and to apply a tree search algorithm towards an identified goal [4, 5]. Well-known proposed variations of this include the roadmap approach in [6], the cell decomposition approach in [7], and the potential field approach in [8]. Typical solution algorithms are Dijkstra and A*, as in [9, 10]. Since the output of these algorithms is a non-smooth path that does not satisfy the kinematic constraints, it usually requires an additional smoothing step, e.g., by numerical optimization [11, 12]. Another common approach is a random search in the space of all kinematically feasible paths, by rapidly exploring random trees [13]. However, there are a number of potential difficulties with using these graph approaches for autonomous driving on highways: The discretization of the coordinate space, the combinatorial explosion of the decision tree (which is a burden for online computation), the poor performance while dealing with a dynamic environment, and the necessity of fixing the start and goal points.

Other path planning algorithms exist that are not based on a discretization of the overseen terrain, for example using Model Predictive Control (MPC) [14, 15]. Such optimization-based methods are often computationally expensive and they can only manage short planning horizons in real-time. However, an effective control strategy on highways requires a long planning horizon, especially in complicated traffic situations, in order to yield a human-like performance. Incorporating this aspect into the decision process is the main motivation behind our path planner.

The main idea of our approach is to consider a finite set of deterministic candidate paths in the form of a decision tree. This set is a small subset of all possible paths due to the restrictions imposed by well-defined driving rules in [16]. Algorithms using motion primitives [17] or probabilistic approaches [18] exploit this structure as well. In order to limit the growth of the decision tree, it is crucial to prune all branches if their paths lead to a situation in which the vehicle is unsafe, or passenger comfort is compromised. Finally, only a small number of feasible paths remain, and the path planner selects the most compatible path with our driving objective (e.g., maximize traveled distance, minimize journey time or fuel consumption).

The proposed path planning algorithm is based on the framework of binary decision diagrams (BDDs) [19, 20]. The BDD is a very intuitive approach, as it attempts to mimic humans' reasoning process about decisions such as lane changes and speed adaptations. The use of BDDs has already been tested with success for robot static obstacle avoidance in [21].

The novelty of our approach is based on the fundamental assumption that there is an 'intelligent' controller for path tracking in place, which is itself predictive and able to avoid obstacles. Hence, neither does the path planner require a high prediction accuracy for the avoidance of collisions, nor a very fast reaction time in the case of dangerous situations. However, the path planner needs to have a long planning horizon and be computationally cheap. In our work, the required 'intelligent' path tracker is implemented using MPC [14].

The remainder of the paper is structured as follows: Section II describes our problem more precisely and explains the hierarchical approach used. Section III provides details of the proposed path planning algorithm. Section IV shows results from simulations and experiments, and concluding remarks are made in Section V.

*Laurène Claussmann is with the Department of Automatic Control, Systems and IT, ENSE3 at Grenoble INP, France (e-mail: laurene.claussmann@grenoble-inp.org)

†Ashwin Carvalho and Georg Schildbach are with the Department of Mechanical Engineering at the University of California Berkeley, USA (e-mail: ashwinc,schildbach@berkeley.edu).

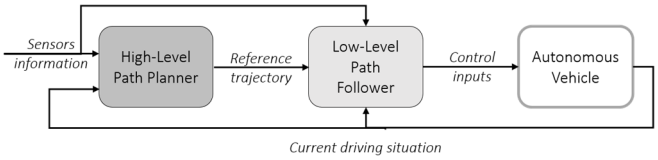

Fig. 1 - Architecture of the hierarchical approach.

II. Problem Description

We use a hierarchical approach to the path planning and control problems for autonomous vehicles on highways, as represented on Fig. 1. The 'high-level' path planner generates a reference trajectory for the controlled vehicle as a function of the current driving situation. The 'low-level' path follower computes the control inputs to the controlled vehicle to closely track this reference trajectory. An overview of each of these subsystems is presented below.

A. Path Following using Model Predictive Control

The problem of controlling an autonomous vehicle to follow a desired path is complicated by (i) the nonlinearity of the vehicle dynamics, and (ii) the presence of state and input constraints due to safety requirements such as collision avoidance. MPC has been shown to be an effective control strategy for autonomous vehicles due to its ability to systematically handle nonlinearities and constraints at the design stage [14, 15].

In MPC, at each time instant, a constrained finite horizon optimal problem is solved to obtain a sequence of inputs. The first element of this sequence is applied to the system, and the process is repeated at the next sampling time. In this work, the cost function of the online optimization problem penalizes the deviation of the predicted controlled vehicle positions from the reference trajectory provided by the path planner. Additional terms are included in the cost function to penalize the control effort and the change in control inputs between consecutive time steps. The vehicle model used for the control design is a nonlinear bicycle model with a linear tire model, where the commanded inputs are the steering and longitudinal acceleration. Collision avoidance constraints are formulated using the approach presented in [22], while the non-convex optimization problem is solved using the methodology presented in [14]. The details are omitted since the focus of this paper is on the path planning algorithm.

The use of a constrained optimal control approach such as MPC has two benefits. Firstly, the use of a high fidelity vehicle dynamics model in the MPC formulation results in planned trajectories that are dynamically feasible and that yield small closed-loop errors. Secondly, the ability to handle collision avoidance constraints introduces safety redundancy in the event that the path planner is unable to generate an obstacle free trajectory.

The main limitation of an optimization based approach such as MPC is the computational complexity of the online optimization problem. This limits the length of the prediction horizon and is a major barrier in applying MPC for path planning. Moreover, the controller is not aware of the overall driving objective of advancing as far as possible on the highway, and the constraints imposed by driving rules. This role is fulfilled by the path planner, which is described next.

B. Path Planning for Autonomous Highway Driving

The proposed approach is based on the intuitive planning that a human driver might apply in a similar fashion. The approach is illustrated by the example in Fig. 2.

When driving on a highway with a slower vehicle in the front, the controlled vehicle evaluates the best option of changing lanes to the left or right, or staying in its lane (3 solid blue arrows in Fig. 2). Considering also a follow-on lane change, the options are limited to at most 9-direction choices (5 dashed green arrows in Fig. 2). Moreover, the driver can choose its planned acceleration profile (e.g., to accelerate or decelerate) and the style of its steering maneuvers (e.g., smooth or aggressive steering). The following are the basic assumptions for the path planner, which are derived from [16] or are common sense.

Assumption 1. The controlled vehicle will keep its current lane, unless there is a reason to perform a lane change (e.g., the current lane is blocked within some critical look-ahead distance).

Assumption 2. The environment evolves deterministically and independently of the controlled vehicle over the planning horizon.

Assumption 3. If the controlled vehicle is in the leftmost lane, an obstacle vehicle approaches it in the same lane with a higher speed from behind, and the right lane is open, then a lane change to the right is initiated.

Assumption 4. When overtaking a slower vehicle in the front, the left lane is preferred to the right lane.

Assumption 2 is not strictly required by our proposed method, and it will be relaxed in future work.

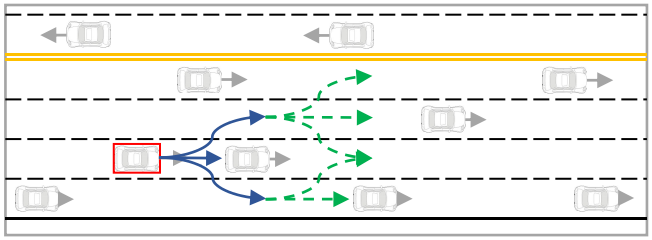

Fig. 2 - An example of a traffic situation on the highway. The small solid grey arrows in front of each vehicles represent their own speed. The red boxed vehicle indicates the controlled vehicle. The solid blue arrows in front of our vehicle are the 3 available directions for a planned path. The dashed green arrows after solid blue arrows constitute the anticipated follow-up directions after the first action.

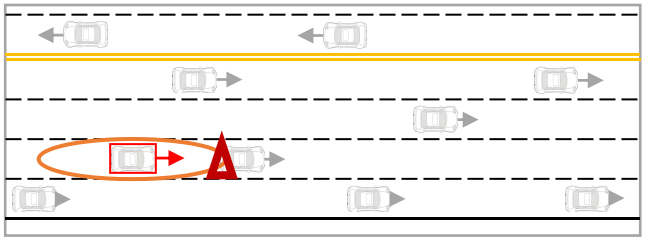

Fig. 3 - An example of a traffic situation on the highway. The small solid grey arrows in front of each vehicles represent their own speed. The red boxed vehicle indicates the controlled vehicle. The orange ellipse around our vehicle constitutes its critical action distance space. The red triangle indicates the penetration of an obstacle into the ellipse, which means a potentially unsafe situation.

III. Algorithm Description

To exploit the special structure of path planning on highways, we propose a tree algorithm with a truth table decision process, which comes under the class of the binary decision diagrams (BDDs) [19, 20]. This section discusses the components of the proposed approach in more detail.

A. Triggering of the Path Planning Algorithm

The algorithm is only triggered when the controlled vehicle is within a critical action distance to another car, which may lead to the path planner initiating an overtake maneuver or a commanded deceleration.

The critical action distance is modeled as an ellipse centered at the controlled vehicle. The ellipse shape is computationally simple in terms of collision detection and has proven to be effective in our simulations. The ellipsoidal axes can be considered as tuning variables for the path planner. They determine how close another car can get (longitudinally and laterally) before the path planner considers taking a new action, as in Fig. 3. They should be chosen large enough for the car to react before a critical situation occurs, but small enough for the car not to perform erratic lane changes. In (1), we choose the major axis to be equal to twice the braking distance (the perception-reaction time is calculated numerically and supposed to be negligible compared to the braking time) and the minor axis to be equal to the vehicle width plus half of the distance between the vehicle's side and the lane border:

$$\text{ellipse:} \begin{cases} \text{major axis} = 2 \cdot \dfrac{\text{vehicle speed}^2}{2 \cdot \text{maximum acceleration}} \\ \text{minor axis} = \dfrac{\text{lane width} + \text{vehicle width}}{2} \end{cases} . \quad (1)$$

A minimal safety ellipsoidal space (half of the critical action distance) is also required by the algorithm to validate a safe lane change (i.e., without any collision all along the predicted trajectory).

Finally, for safety and comfort reasons the controlled vehicle is required to complete a lane change, once it has been initiated.

Algorithm 1 – Trigger of Path Planner

Observe the critical action distance of the car

If *no car is within the critical distance* **then**

The reference path is "keep the lane, at the speed limit"

Else (another car is within the critical distance)

If *our car is currently in a lane change* **then**

Wait until our car has reached the new lane

Else

Calculate a new path

End if

End if

B. Candidate Path Construction

The path planner considers a finite number n_{cand} of longitudinal acceleration and lane changing profiles over the planning horizon. Let s denote the longitudinal position of the controlled vehicle along the road and e_y denote its lateral position with respect to the road left border. A lane change corresponds to a step change in e_y, in which case the basic reference path is modeled by a parameterized sigmoid function [23]:

$$e_y = e_{y,0} + \frac{b}{1 + \exp\left[-a \cdot (s - c)\right]} . \quad (2)$$

In (2), $e_{y,0}$ is the current lateral position of the controlled vehicle at the beginning of the maneuver, b is the difference between $e_{y,0}$ and the desired lateral position at the end of the maneuver, a is a slope parameter, and c is a distance delay. The parameters a and c are tuning parameters for driver comfort, which depend on the vehicle's velocity.

The corresponding acceleration profiles are based on experiments and intelligent driver models [24]. The profiles are designed in accordance with the longitudinal and lateral acceleration constraints set by the tire-road friction limits and the speed limit. The path planner algorithm also prefers profiles which are smoother and better account for the well-being of passengers. The candidate profiles are then sorted in a matrix used for the truth table process (*first order profile matrix*).

C. Binary Decision Tree

The binary decision tree aims at finding all candidate paths that are collision-free over the planning horizon T_P, which is chosen to be much larger than the MPC prediction horizon. It then selects the most appropriate candidate path based on given optimality criteria.

The necessary information for the path planner about each obstacle $k = 1,\ldots,K$ consists of its longitudinal and lateral positions s_k and $e_{y,k}$, respectively, its current lane n_k, its speed v_k, and its dimensions; see Fig. 4. The lanes are indexed by n and are numbered from left to right, i.e., the leftmost lane is given by $n = 1$.

Considering the approach of maximizing the traveled distance, we consider the following two criteria (see Fig. 5): First, s_C, which is the longitudinal distance to the first fictional collision point. This fictional collision point is constructed based on a considered candidate path and the

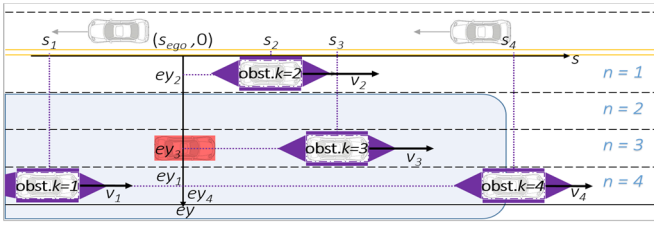

Fig. 4 - Environment definition. The black arrows in front of each vehicles represent their own speed. The red boxed vehicle indicates the controlled vehicle. The blue rectangular area is the sensors' analyzed region. The purple rectangular and triangular shapes around the obstacle vehicles are required by the MPC algorithm [14].

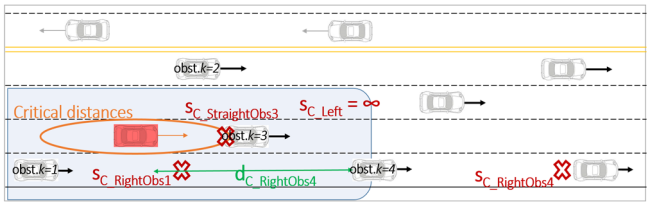

Fig. 5 - Illustration of the two decision criteria used: s_C and d_C.

predicted trajectories of all other vehicles $k = 1,\ldots,K$ in the sensors' analyzed region. Second, d_C, which is the sum of the distances of the controlled car to the obstacle corresponding to the fictional collision point, is computed as:

$$d_C = \sum_{t=0}^{T_C} \left| \mathbf{s}_t - \mathbf{s}_{t,k} \right| . \quad (3)$$

In (3), T_C is the fictional collision time, t represents each sampling time step from 0 to T_C, s_t and $s_{t,k}$ are the positions of the controlled vehicle and obstacle k at t, respectively.

Assumption 5. The bigger s_C for a particular path, the further the controlled vehicle will advance in the traveled direction.

Assumption 6. The bigger d_C for a particular path, the safer is the chosen trajectory for the controlled vehicle.

The BDD returns the best combination of lane change decisions and longitudinal acceleration profiles within two decision levels, as illustrated in Fig. 6.

The first level of the decision diagram starts with three choices of the main direction for the new path: right lane change ('Right'), left lane change ('Left') or keep the same lane ('Straight'). Only the feasible directions are further analyzed. For example, if the controlled vehicle is on the leftmost (rightmost) lane, a left (right) lane change is impossible. The n_{cand} longitudinal acceleration profiles are generated for each feasible main direction and the corresponding s_C and d_C are computed. The path planner algorithm requires that the controlled vehicle can reach the centerline of the desired lane without a collision; otherwise the path is dismissed.

For each direction (Right, Left, Straight), the best acceleration profile-path (according to given criteria) is then selected from all collision free lane change paths. In our paper, the best path is the one with the largest value of s_C; in case of a tie, the path with the largest value of d_C is selected; in case of a second tie, the path with the best position in the *first order profile matrix* is chosen. Hence the outcome of the first level consists of at most three candidate paths, which prevents a combinatorial explosion of the decision tree.

The second level of the decision tree repeats the same process as the first level, for each of the three (or less) branches corresponding from the first level. Thus, the algorithm produces at most 9 candidate paths as the output of the second level.

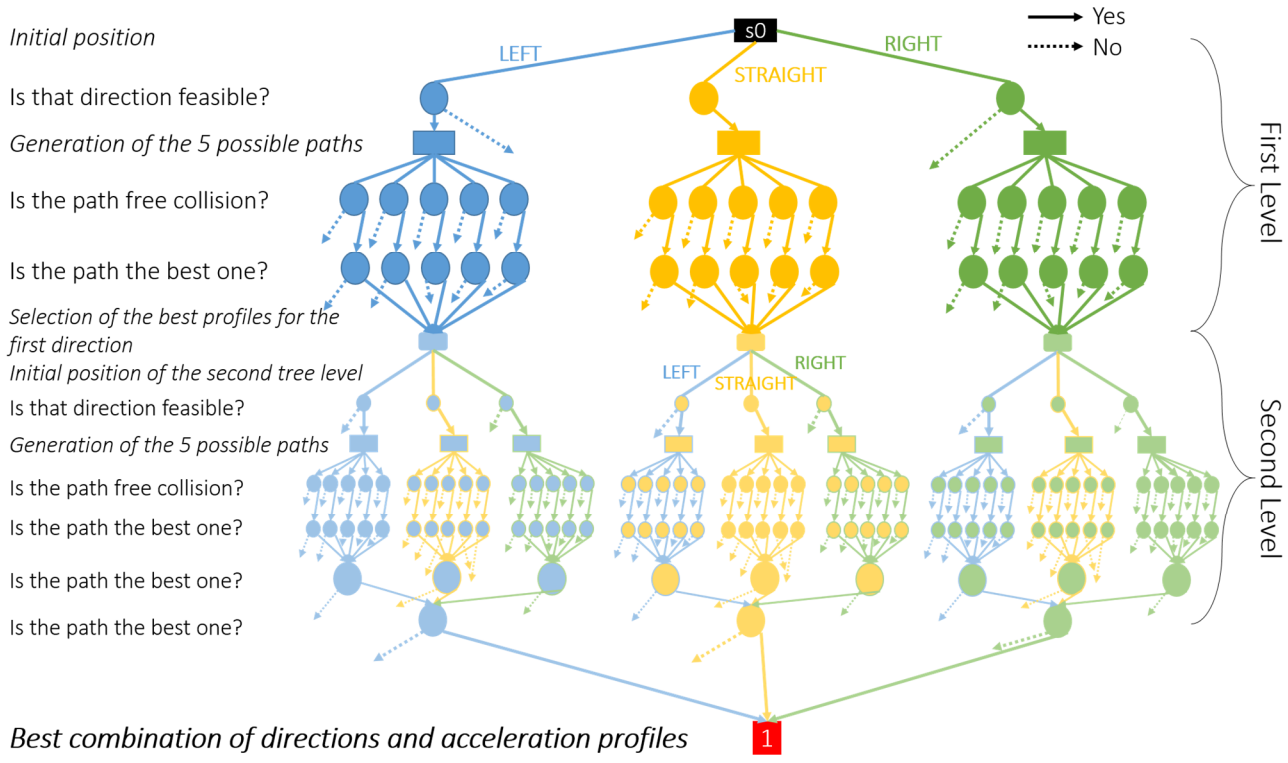

Fig. 6 - The Binary Decision Diagram with 5 n_{cand}: The black box "s0" is the longitudinal position of the controlled vehicle when the path planner is triggered. The first tree-level consists of a choice "Left (left blue forms) – Straight (center yellow forms) – Right (right green forms)". The first branch ends with the 3 best acceleration profiles (rectangular forms). These 3 boxes are the initial position of the second level. The same scheme as for the first level is repeated for each of the 3 boxes of the second level. The circles are the decision nodes linked with the Boolean decision questions on the left. A 'yes' answer corresponds to a solid arrow, whereas a 'no' answer corresponds to a dotted arrow. The 'no' answers lead directly to a 0-terminal (not represented here). The full 'yes' answers lead to the 1-terminal schematized in the '1' red box.

Considering the main direction, the candidate second directions are compared with the truth table decision maker described in subsection III.D. At the end of this step, we get at most 3 final candidate paths. The final candidate paths are then compared with the truth table decision maker described in subsection III.D. At the end of this step, we get the best combination of directions and longitudinal acceleration profiles for every initial situation.

The path planner provides the coordinates of the chosen path as a reference to the MPC-based path follower. The longitudinal speed profile along the path is also provided. When the lane change maneuver is completed, the path follower keeps the vehicle close to the lane centerline at the desired speed till the path planner initiates the next maneuver.

D. Truth table decision maker

The truth table decision maker returns the best direction according to the two criteria s_{C_D1D2} and d_{C_D1D2} analysis, as defined in Tables I and II.

Assumption 7. The fewer lane changes of the controlled vehicle are planned, the safer is the chosen trajectory.

A first truth table compares the values of the first criterion s_{C_D1D2} (cf. Table I) for the feasible directions. If there is only one maximum of s_C, the directions combination associated is the best one. If there are 2 or 3 candidates for the maximum, Table II, which compares the second criterion d_{C_D1D2}, is applied to the candidate in the same way as Table I except in case of more than one maximum for d_{C_D1D2}, to respect the Assumption 4, the straight direction is favored, then the left and the right.

TABLE I. TRUTH TABLE FOR THE FRIST CRITERION

Is that value the maximum? 1 = Yes | 0 = No			
$s_{C\_D1Straight}$	s_{C_D1Left}	$s_{C\_D1Right}$	***Choice***
0	0	0	*Impossible*
0	0	1	Go Right
0	1	0	Go Left
0	1	1	Table II L-R
1	0	0	Go Straight
1	0	1	Table II S-R
1	1	0	Table II S-L
1	1	1	Table II S-L-R

TABLE II. TRUTH TABLE FOR THE SECOND CRITERION (TABLE II L-R)

Is that value the maximum?		
d_{C_D1Left}	$d_{C\_D1Right}$	***Choice***
0	0	Go Left
0	1	Go Right
1	0	Go Left
1	1	Go Left

IV. SIMULATION AND EXPERIMENTAL RESULTS

In this section, we present some simulation and experimental results to demonstrate the working of our controller. The results show that our path planner effectively handles regular traffic situations and that it can be run in real-time on embedded hardware, paired with an MPC-based controller [14].

A. Simulation Setup Description and Results

The scenario for the simulation study is depicted in Figs. 7-8. The values used to define the ego vehicle and obstacles are reported in Table III. We consider a 3-lane road over 2250m, our vehicle, 3 moving obstacles corresponding to a regular traffic and a work zone (static obstacle). The planning horizon of the path planner is 15s, while that of the MPC is 1s.

TABLE III. INITIALIZED PARAMETERS OF THE SIMULATION SCENARIO

Vehicles	**Initial longitudinal position s (m)**	**Initial lane #**	**Speed (m/s)**	**Length (m)**	**Width (m)**
controlled vehicle	300	2	30	2.9	1.625
obst. $k = 1$	600	2	20	3	2
obst. $k = 2$	520	1	30	3	2
obst. $k = 3$	1200	2	20	3	2
obst. $k = 4$	2500	3	0	1000	3

The situation around the vehicle is safe, until the controlled vehicle gets too close to the obstacle $k = 1$. This triggers the path planner. Both left and right lanes are free for a lane change, so the path planner calculates $s_{C\_Straight}$, s_{C_Left} and s_{C_Right}. The second level of the tree is then tested. For the 'First Left Choice', the only available second directions are 'Straight' or 'Right' in order to overtake the slower obstacle. But for this last path, the minimum safety space (inside circle) is not completely respected during the lane change. The farthest the controlled vehicle can advance is to $s_{C\_LeftStraight}$. For the 'First Right Choice', the only available directions are 'Straight' or 'Left' in order to overtake the slower obstacle. For this last path, the safety space is completely respected during the lane change, but it involves a second lane change, whereas 'Straight' is safe too. The farthest the controlled vehicle can advance is to $s_{C\_RightStraight}$. The path planner selects the combination of directions 'Right/Straight' for the next high-level time horizon (Fig. 7). Only the first direction is applied, the second one is an anticipated direction. When the first lane change is finished (Fig. 8), the controlled vehicle is on lane 3 but obstacle $k = 4$ is on lane 3. As the controlled vehicle gets too close to that obstacle, the path planner is triggered again. According to the same process as before, the direction 'Straight' and 'Left' are evaluated $s_{C\_Straight}$ and s_{C_Left}. With main direction 'Left', the second feasible directions are 'Left' and 'Straight'. In our simulation, the best path is the combination of 'Left/Left'.

To ensure that the planned path respects the minimal safety ellipsoidal space, we look at the longitudinal and lateral distances between the ego vehicle and the obstacles, as a function of the longitudinal position of the ego vehicle along the road (Fig. 9).

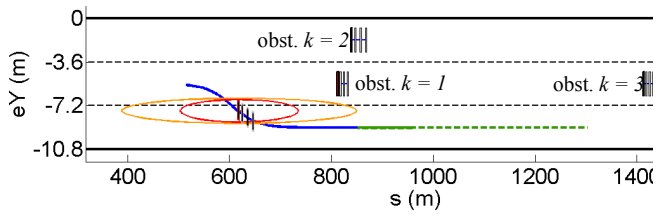

Fig. 7 - Simulation result 1/2. First level is in solid blue line, second level in dashed green line, critical action distance is the outer orange ellipse and minimal safety ellipsoidal space is the inner red ellipse.

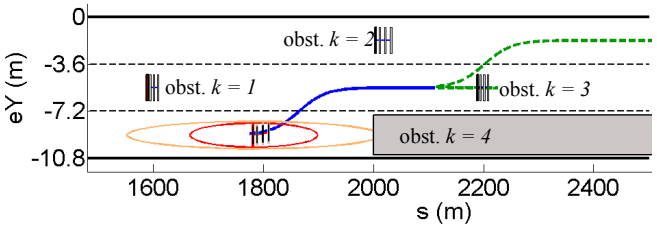

Fig. 8 - Simulation result 2/2. First level is in solid blue line, second level in dashed green line, critical action distance is the outer orange ellipse and minimal safety ellipsoidal space is the inner red ellipse.

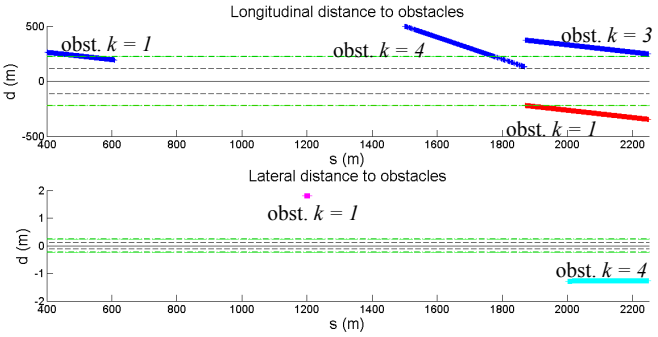

Fig. 9 - Simulation result. Critical action distance is represented by the outer dash-dotted green strip and minimal safety ellipsoidal space is the inner dashed black strip. The other colored lines are the distance between the ego vehicle and obstacles detected in the sensors' area (blue = front obstacle; red = rear; magenta = left; cyan = right). If one of the colored lines passed through the outer dash-dotted green strip, the path planner is triggered. No obstacle should enter the inner dashed black strip, otherwise the minimal safety space is not respected.

In Fig. 9, obstacle $k = 1$ enters first the critical action distance space (outer dash-dotted green strip) in front of the controlled vehicle: the path planner is triggered (result in Fig. 7). When obstacle $k = 4$ enters the path planner's strip as in Fig. 8, the left change path is selected because the path is safe: both front and rear ($k = 4,3,1$) obstacles do not enter the minimal safety ellipsoidal space (inner dashed black strip).

B. *Experimental Setup Description and Results*

The experiments were performed on a Hyundai Azera 2011 test vehicle at the Hyundai Kia California Proving Grounds, using the straight stability road (4-lane straight road). An Oxford Technical Solutions RT2002 sensing system is used to measure the vehicle position and orientation in an inertial frame, and the velocity in the body frame. The RT2002 comprises of a differential GPS, an IMU and a DSP. The path planning and following algorithms are run on a dSPACE MicroAutoBox II which consists of a 900 MHz PowerPC processor. The goal of the experimental tests is to show the real-time feasibility of our path planner combined with the MPC, and to evaluate the wellbeing of the car's passengers. The planning horizon of the path planner is 15s, while that of the MPC is 1s. The control actions are executed at 20 Hz.

To ensure the safety of the car and passengers during the 2 experiments, we included an additional safety distance of 20m at the front and back of each ego vehicle and obstacles, and we increased the critical action distance to one and a half times its original value.

The first experimental scenario is a straight 4-lane road with the initialization parameters given in Table IV.

TABLE IV. INITIALIZED PARAMETERS OF THE EXPERIMENT SCENARIO 1

Vehicles	Initial longitudinal position s (m)	Initial lane #	Speed (m/s)	Length (m)	Width (m)
controlled vehicle	0	4	15	2.9	1.625
obst. $k = 1$	250	4	5	3	2
obst. $k = 2$	350	3	5	3	2

The potential collision with obstacle $k = 1$ is on the rightmost lane at 250m. Hence, only two feasible paths are tested, lane following and left lane change. As the left lane change is collision free, it corresponds to the best path to follow. With this choice of direction, the controlled vehicle is predicted to collide with obstacle $k = 2$ at 400m. To avoid the collision, either a left lane change or right lane change is feasible. As specified in Assumption 4, the left lane change is favored. The expected behavior is thus 'Left/Left' as seen in Fig. 10.

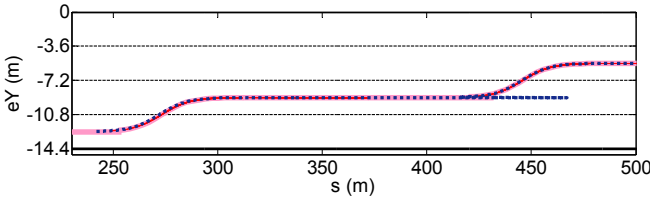

Fig. 10 - Experimental scenario 1. The dotted blue line is the reference path, the solid pink line is the driven path of the controlled vehicle.

The second experimental scenario is a straight 4-lane road with the initialized parameters of Table V.

TABLE V. INITIALIZED PARAMETERS OF THE EXPERIMENT SCENARIO 2

Vehicles	Initial longitudinal position s (m)	Initial lane #	Speed (m/s)	Length (m)	Width (m)
controlled vehicle	200	1	15	2.9	1.625
obst. $k = 1$	0	1	25	3	2

According to the specified rules for lane changing (Assumption 3), the controlled vehicle should change lanes to the right as soon as obstacle $k = 1$ comes too close to it from behind (which is theoretically at 375m) and a right lane change is feasible. This situation is respected in our scenario 2 (Fig. 11). Thus, the expected behavior of the controlled vehicle is 'Right/Straight'. During the experiment, the obstacle enters the critical action distance at 375m along the road, and the directions' combination chosen was 'Right/Straight' at constant speed.

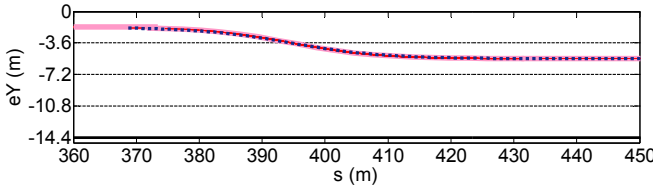

Fig. 11 - Experimental scenario 2. The dotted blue line is the reference path, the solid pink line is the driven path of the controlled vehicle.

As seen in Figs. 10 and 11, the path follower is able to closely track the reference provided by the path planner. This validates the choice of the sigmoid function in [23] as a dynamically feasible path for describing a lane change maneuver. The online computation times for the proposed path planning algorithm and that for the combined path planner and MPC-based controller are shown in Table VI. This shows that our algorithm is fast enough to deal with a dynamic environment.

TABLE VI. COMPUTATION TIME OF THE PROPOSED ALGORITHM

Computation time (ms)	**Path Planner (PP)**	**PP + MPC**
Mean value	2	7.5
Max value	2.5	11.6

Besides, the passengers of the vehicle during the test did not feel any difference between the driving behavior of the human driver and the autonomous vehicle, which confirms the choice of the curvature of the path in order to respect the comfort of passengers.

V. CONCLUSION

We designed a path planner based on BDD, which is simple but robust enough to deal with a dynamic environment, traffic rules and the well-being of passengers. Moreover, it is optimized to reduce computational effort: a new path is calculated only when necessary and the combinatorial explosion is limited in the development of the tree algorithm. Thus we have presented a new method for path planning on a highway, which is practically useful for autonomous cars. Future work involves extending the environment model to interactive and stochastic scenarios as well as performing more complex (including car pool lanes, merging lanes, obstacles' changing speeds and directions) and dealing with several autonomous vehicles at the same time (communication and priority decision process).

ACKNOWLEDGMENT

The authors gratefully acknowledge financial support from the Hyundai Center of Excellence at the University of California at Berkeley. Moreover, the authors would like to thank Stéphanie Lefèvre and Tammo Zobel for their help with the test vehicle, and Hyundai Kia Motor Company for providing us with the test facilities.

REFERENCES

[1] J.-P. Laumond, S. Sekhavat, and F. Lamiraux. Guidelines in nonholonomic motion planning for mobile robots. In *Robot motion planning and control*, pages 1–53. Springer-Verlag, 1998.

[2] S. M. LaValle. *Planning algorithms*. Cambridge University Press, Cambridge, U.K., 2006.

[3] S.D. McKeever. *Path planning for an autonomous vehicle*. Master's thesis, Massachussetts Institute of Technology, 2000.

[4] T. Ersson and X. Hu. Path planning and navigation of mobile robots in unknown environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Maui (HI), USA, 2001.

[5] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel. Practical search techniques in path planning for autonomous driving. In *First International Symposium on Search Techniques in Artificial Intelligence and Robotics*, Chicago (IL), USA, 2008.

[6] P. Bhattacharya and M.L. Gavrilova. Roadmap-based path planning - using the voronoi diagram for a clearance-based shortest path. *IEEE Robotics Automation Magazine*, 15(2), pages 58–66, 2008.

[7] S. Garrido, L. Moreno, M. Abderrahim, and F. Martin. Path planning for mobile robot navigation using voronoi diagram and fast marching. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, 2006.

[8] Y.K. Hwang and N. Ahuja. A potential field approach to path planning. *IEEE Transactions on Robotics and Automation*, 8(1), pages 23–32, 1992.

[9] D. K. Fan and P. Shi. Improvement of Dijkstra's algorithm and its application in route planning. In *7th International Conference on Fuzzy Systems and Knowledge Discovery*, Yantai, China, 2010.

[10] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, and S. Thrun. Anytime dynamic A*: An anytime, replanning algorithm. In *International Conference on Automated Planning and Scheduling*, Monterrey (CA), USA, 2005.

[11] D. Ferguson and A. Stentz. Field D*: An interpolation-based path planner and replanner. In Sebastian Thrun, Rodney Brooks, and Hugh Durrant-Whyte (editors), *Robotics research*, Volume 28 of Springer Tracts in Advanced Robotics, pages 239–253. Springer, Berlin, 2007.

[12] Y. Wang and G.S. Chirikjian. A new potential field method for robot path planning. In *IEEE International Conference on Robotics and Automation*, San Francisco (CA), USA, 2000.

[13] S. M. Lavalle. *Rapidly-exploring random trees: A new tool for path planning*. Technical report, Iowa State University, 1998.

[14] A. Carvalho, Y. Gao, A. Gray, H.E. Tseng, and F. Borrelli. Predictive control of an autonomous ground vehicle using an iterative linearization approach. In *16th International IEEE Conference on Intelligent Transportation Systems*, Den Haag, Netherlands, 2013.

[15] Y. Gao, A. Gray, J. V. Frasch, T. Lin, E. Tseng, J. K. Hedrick, and F. Borrelli. Spatial predictive control for agile semi-autonomous ground vehicles. In *Advanced Vehicule Control Conference*, Seoul, South Korea, 2012.

[16] California Driver Handbook, English version, 2014. Available online at www.dmv.ca.gov.

[17] A. Gray, Y. Gao, T. Lin, J. K. Hedrick, H. E. Tseng, and F. Borrelli. Predictive control for agile semi-autonomous ground vehicles using motion primitives. In *American Control Conference*, Montreal, Canada, 2012.

[18] L. Li and F.-Y. Wang. The automated lane-changing model of intelligent vehicle highway systems. In *5th IEEE International Conference on Intelligent Transportation Systems*, Singapore, Singapore, 2002.

[19] S. Sanner. *Decision diagrams in discrete and continuous planning*. Technical report, Australian National University, 2012.

[20] S. Balinova. *Planning based on model checking*. Technical report, Bamberg University, 2010.

[21] A. T. Rashid, A. A. Ali, M. Frasca, and L. Fortuna. Path planning with obstacle avoidance based on visibility binary tree algorithm. *Robotics and Autonomous Systems*, 61(12), pages 1440-1449, 2013.

[22] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel. Motion planning with sequential convex optimization and convex collision checking. *The International Journal of Robotics Research*, pages 1-20, 2014.

[23] M. Arbitmann, U. Stahlin, M. Schorn, and R. Isermann. *Method and device for performing a collision avoidance maneuver*. US Patent App. 12/047,467, 2008.

[24] C.J. Van Leeuwen. *Driver modeling and lane change maneuver prediction*. Master's thesis, University of Groningen, 2010.