Optimal Longitudinal Control Planning with Moving Obstacles

Jeff Johnson and Kris Hauser

***Abstract*— In urban road networks intelligent road vehicles must solve challenging optimal control problems in real-time to navigate reliably around moving obstacles. We demonstrate a complete planner that computes collision-free, optimal longitudinal control sequences (acceleration and braking) using a novel visibility graph approach that analytically computes the reachable subset of path-velocity-time space. We show that our method plans over an order of magnitude faster than previous approaches, which makes it scalable and fast enough (tenths of a second on a PC) to be called repeatedly on-line. We demonstrate applications to autonomous driving and vehicle collision warning systems with up to 20 moving obstacles.**

I. INTRODUCTION

The Federal Highway Administration in the United States notes that the frequency of automobile collisions is directly related to the number of *conflict points* [1], which are points in a vehicle's path that are crossed by the paths of obstacles, such as pedestrians, bicyclists, and other vehicles. Complex urban intersections may involve dozens of conflict points and require an intelligent vehicle to simultaneously avoid leading vehicles, merging vehicles, cross-traffic, and pedestrians. While specialized collision avoidance techniques have addressed specific conflict types, such as following lead vehicles [2], lane merging [3], and cross-traffic [4], there is very little work on general-purpose techniques that address heterogeneous conflict types.

To address this problem we have developed a planner that extends an analytical approach previously developed for negotiating cross-traffic [4]. Given a desired vehicle path and estimates of future obstacle behavior, our planner computes a *visibility graph* that represents the set of all possible path/velocity/time (PVT) states that are reachable via a collision-free longitudinal control sequence. The optimal control sequence is extracted from this graph (Fig. 1).

Our approach is, to our knowledge, the first exact algorithm to handle both acceleration and velocity bounds in the presence of moving obstacles in guaranteed polynomial time. The planner is also complete, which allows it to solve both the planning problem (*how* to traverse a given path) and the decision problem (*whether* a given path is traversable). This completeness allows it to be used both in lower level vehicle control routines and in higher level planning routines, such as collision avoidance and warning systems. As such, it represents a contribution to both the fields of intelligent transport systems and autonomous vehicles.

Our planner can handle arbitrary vehicle paths, polygonal vehicle and obstacle models, and arbitrary velocity profiles,

Jeff Johnson and Kris Hauser are with the School of Informatics and Computing, Indiana University {jj56,hauserk}@indiana.edu

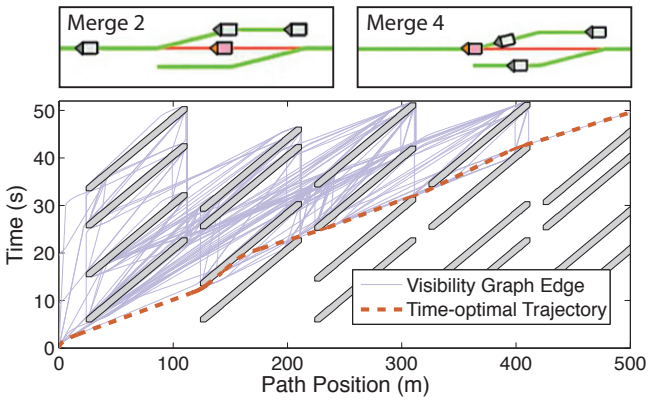


Fig. 1: Visibility graph of scenario with 20 vehicles merging onto/off of the driver's path. The vehicles also share portions of the driver's path, leading to diagonal PT Obstacles. The supplemental video contains a simulation of this scenario.

and it also naturally handles uncertainty by "growing" the obstacles according to confidence bounds on their future behavior. Our method is over an order of magnitude faster than previous approaches and handles dozens of moving obstacles in tenths of a second on a standard PC. We demonstrate its application in a simulated urban intersection involving pedestrians, bicyclists, and automobiles as well as merging with cross-traffic on a rural highway.

II. RELATED WORK

Dynamic navigation among moving obstacles is a challenging problem with a long history. In one class of problems, the vehicle has choice over spatial as well as velocity controls. Unfortunately, this problem is computationally intractable: even when obstacles have known trajectories, the planar motion planning problem among constant-velocity moving obstacles is PSPACE-hard [5]. More recently, randomized techniques have been used. Kindel, et. al. [6] employed a variant of Probabilistic Roadmaps [7] with fast replanning to plan arbitrary trajectories in the presence moving obstacles and sensor uncertainty. The MIT team [8] for the 2007 DARPA Urban Challenge used a real-time implementation of Rapidly-exploring Random Trees [9] in order to plan vehicle trajectories. Although randomized methods achieve tractability, they sacrifice hard guarantees on completeness and optimality, which can be problematic for road vehicles that must operate near 100% reliably even in heavily crowded environments.

A second class of problems is *longitudinal control planning*. This approach assumes a structured road network

with only a handful of paths available, and assumes that the vehicle may steer along paths with reasonably high accuracy [10], [11], [12]. By decoupling spatial path planning and velocity control problems, the problem becomes more tractable. The Carnegie-Mellon team in the 2007 DARPA Urban Challenge exploited path/velocity decomposition for road lane navigation by computing optimal paths based on the centerline of the road lanes [13]. Their method proposes a set of candidate trajectories along the optimal path and chooses the best according to various metrics, including how well they avoid static and dynamic obstacles. This method, however, offers no guarantees of a safe solution.

Rigorous approaches to solving the longitudinal control planning problem include the *visibility graph* method [14] or explicit search over a discretization of state space [15]. Our method is also a visibility graph that addresses the major shortcoming of the previous method [14] by incorporating acceleration bounds. Unlike explicit discretization [15], our method is analytical and runs in polynomial time, making it fast enough to be used for real-time replanning.

III. PROBLEM DEFINITION

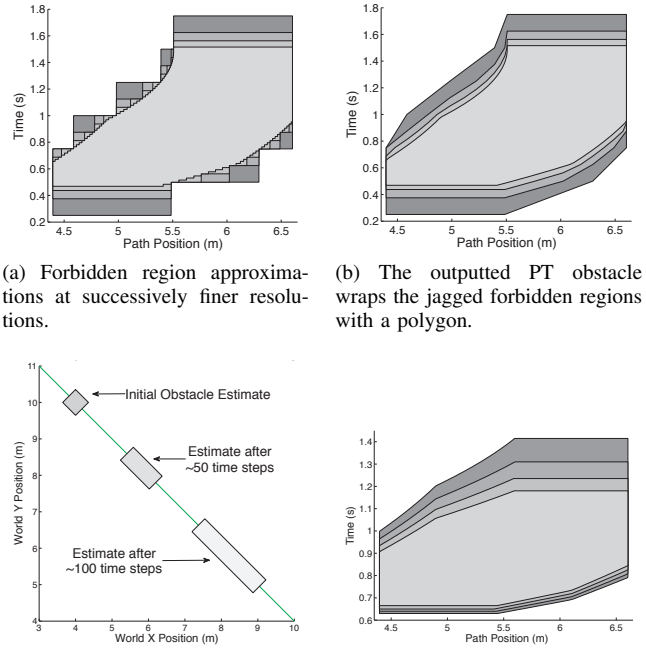
The vehicle is assumed to travel along a known path and to sense objects in its environment and estimate their intended behavior over a fixed time horizon, e.g., using vision, radar, or inter-vehicle communication. The planner is then asked to solve a longitudinal control problem to define the vehicle's future trajectory over this horizon (in the case of an intelligent vehicle) or to deliver a collision warning (for a driver assistance system). The planner is designed to be invoked repeatedly in a model predictive control-like scheme to advance the horizon and respond to changing sensor input. We define the planner's assumptions below.

Inputs. Our planner takes as input the robot R's arc-length parameterized path $P_R(p) : [0, p_{max}] \mapsto \mathcal{C}$, and a list of n obstacles O_i along with their predicted paths $P_{O_i}(p)$, $i = 1, \ldots, n$. All vehicles are modeled as polygons that translate and rotate as they slide along piecewise linear paths. We assume the orientation of a vehicle at any position p along its path is always tangent to the path, and hence its world-space layout is entirely determined by p. Uncertain obstacle behavior is handled by specifying an interval of path positions $p_i(t) \in [\underline{p}_i(t), \overline{p}_i(t)]$ at which obstacle i might lie at time t. We also make the simplifying assumption that obstacle behavior is independent of driver behavior.

Dynamically feasible PVT trajectories. The planner computes a continuous curve in the *path-velocity-time* (PVT) state space, in which states $x = (p, v, t)$ consist of a path position $p \in [0, p_{max}]$, velocity v, and time $t \in [0, t_{max}]$. Dynamic constraints include velocity and acceleration bounds:

$$\begin{aligned} \dot{p} &= v \\ v &\in [\underline{v}, \overline{v}] \ : \ \underline{v} \geq 0 \\ \dot{v} &\in [\underline{a}, \overline{a}] \ : \ \underline{a} < 0 < \overline{a} \end{aligned}$$

A trajectory $x(t) = (p(t), v(t), t)$ defined over $t \in [a, b]$ that satisfies these conditions for all $t \in [a, b]$ is called *dynamically feasible*.



(a) Forbidden region approximations at successively finer resolutions.

(b) The outputted PT obstacle wraps the jagged forbidden regions with a polygon.

(c) Example of growing uncertainty in an obstacle's swept volume as time progresses.

(d) PT obstacles (computed at a fixed resolution) for increasing obstacle behavior uncertainty.

Fig. 2: Illustrating PT obstacle construction.

PT obstacles. Each obstacle O_i imposes a forbidden region CO_i in the *path-time* (PT) plane that corresponds to the (p, t) points that would cause the driver and obstacle geometry to intersect at a given path position and time [15]. With uncertain obstacle behavior we take PT obstacles to be the union of obstacles over all possible path positions $p_i(t) \in [\underline{p}_i(t), \overline{p}_i(t)]$. A trajectory $x(t)$ is *collision free* if $(p(t), t) \notin CO_i$ for all $i = 1, \ldots, n$ and $t \in [a, b]$, and it is *feasible* if it is both dynamically feasible and collision free.

Boundary conditions. The planner must generate a path from the initial state $x(0) = (0, v_0, 0)$ where v_0 is the vehicle's current velocity to one of two goal cases:

1) Successful navigation: $p(t_{end}) = p_{max}$ and $v(t_{end}) \in [\underline{v}_{goal}, \overline{v}_{goal}]$ for some $t_{end} \leq t_{max}$. We allow a range of goal velocities to obey bounds on speed limits.
2) Premature stop: $p(t_{max}) < p_{max}$ and $v(t_{max}) = 0$. This case can occur when lead vehicles are stopped.

The planner will output the Case 1 solution of minimum time if one exists, or the Case 2 solution of furthest progress.

IV. PLANNING SYSTEM

The planner consists of two parts. First, it computes an approximation to the PT obstacles. Then, it builds a visibility graph in PVT space and constructs the optimal trajectory by searching the graph.

A. PT Obstacle Construction

It is challenging to compute the boundaries of PT obstacles exactly because they may be arbitrarily curved. A simple approximation technique might build a grid in PT space

with resolution τ and test each cell for collision, but this requires discretization in two dimensions and a cost of $O((t_{max}p_{max})/\tau^2)$. Instead, our approach discretizes only the time dimension and *analytically* computes forbidden intervals in the path dimension.

To compute the PT obstacle corresponding to a given world obstacle O, consider the time dimension of the PT plane discretized into a uniform grid $0, \tau, 2\tau, \ldots, t_{max}$. Our algorithm uses a forward simulation of obstacle motion to compute the left and rightmost extent of CO_i swept out within each horizontal strip $t \in [k\tau, (k+1)\tau]$ in the PT plane, resulting in a conservative *forbidden rectangle* $[a_k, b_k] \times [k\tau, (k+1)\tau]$. Fig. 2a shows PT obstacles constructed at progressively finer resolutions. The rectangles are then wrapped with a polygon to smooth their jagged edges as shown in Fig. 2b.

With a computational cost of $O(t_{max}/\tau)$ per obstacle this technique leads to significant savings over a naïve discretization of both dimensions, and the approximation approaches the true PT obstacle as $\tau \to 0$.

B. Visibility Graph Planner

The planner then proceeds from the observation that any time-optimal trajectory will either connect trivially to the goal, or be tangential to a forbidden region on the PT plane. Because there will always be a time-optimal trajectory if the path is traversable, searching among tangential trajectories is guaranteed to find a solution if one exists. To search tangential trajectories, the planner computes sets of reachable velocities at each PT obstacle vertex by constructing a visibility graph.

The computation of reachable velocity sets at each vertex relies on the properties of *homotopy channels* within PT space. A homotopy channel H is a region of PT space given by upper and lower bounds $l(t) \leq p(t) \leq u(t)$. The mathematical principle behind our approach is that all extremizing trajectories from one vertex to another through a single homotopy channel are a combination of bang-bang motions in free-space and portions that are tangent to $l(t)$ and $u(t)$ [16]. In [4] we proved that the set of reachable velocities from a starting state $x_1 = (p_1, v_1, t_1)$ through a single H and reaching a goal PT point (p_g, t_g) is convex in the presence of rectangular PT obstacles, and we have since extended this approach to handle polygonal PT obstacles. Hence, in order to define the reachable velocity set at a vertex, it suffices to find the velocity-extremizing trajectories in H ending at (p_g, t_g). Visibility graph construction proceeds by iteratively propagating reachable velocity sets computed with extremizing trajectories forward through the PT obstacle vertices. A polynomial-time algorithm for velocity set propagation is given in [4].

Once the visibility graph is computed, a time-optimal trajectory is recovered by tracking backwards from the goal vertex, as in Fig. 1. Note that the graph is a complete representation of *all* feasible trajectories, so it may also be possible to optimize other objective functions like maximum safety. The complexity of the planner is polynomial in the total number of PT obstacle vertices, and in practice can re-plan at up to 10Hz with up to 20 PT obstacles.

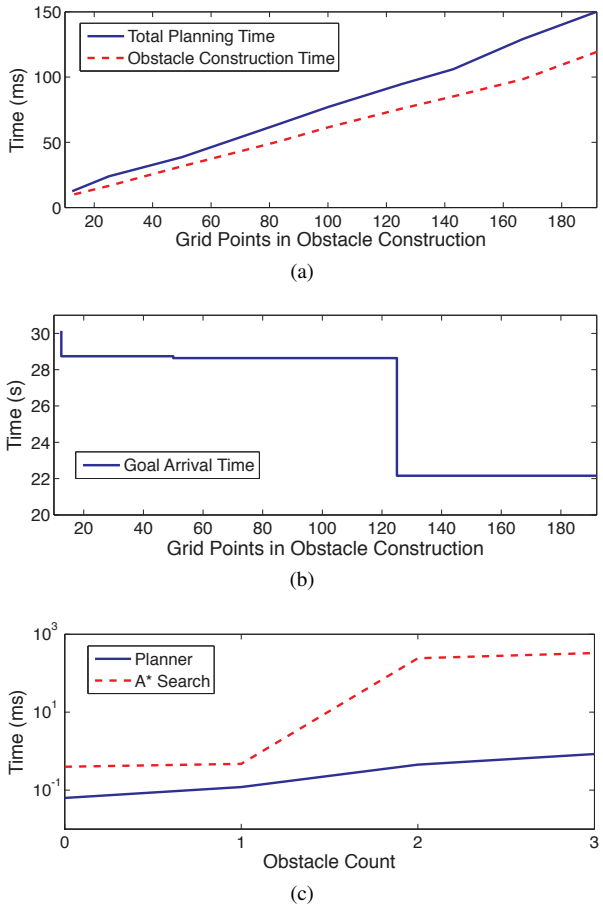
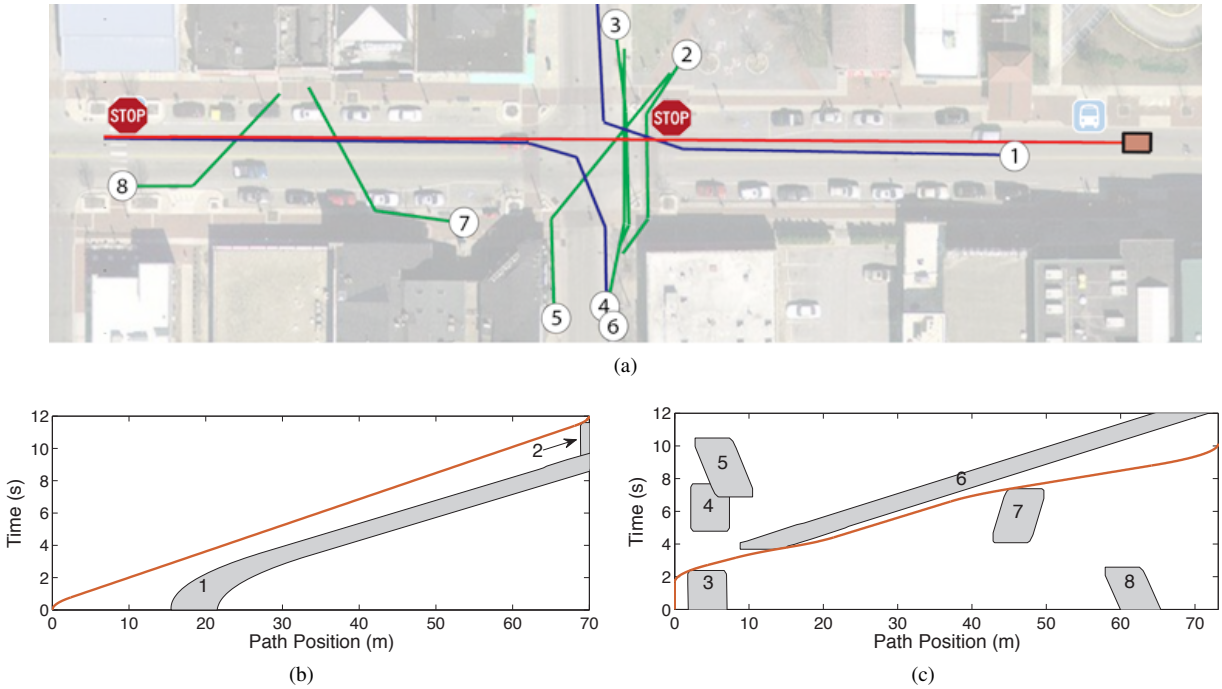


Fig. 3: (a) Running times for obstacle construction and planning according to increasingly fine obstacle discretization. (b) Trade-off between obstacle discretization and optimality of plan. (c) Comparing our planner against A^* search for increasing obstacle counts in a toy scenario. (Note the logarithmic scale on the time axis).

C. Empirical Performance

Fig. 3 shows empirical performance of PT obstacle construction and visibility graph construction for a scenario similar to that in Fig. 4. Results are averages of 10 runs on a single core of a 2.3GHz PC. In Fig. 3a the planner is run with varying levels of discretization in PT obstacle construction, showing a roughly linear relationship. Fig. 3b shows how discretization affects the optimal goal arrival time. At coarse discretizations, narrow passageways in PT space are occluded and the planner goes around them. At finer discretizations, these channels open up and the planner finds a faster route.

Fig. 3c compares our method to the A^*-based search method of [15]. Both planners were run on a simple scenario with zero to three obstacles. Velocity was constrained to be within $[0, 20] m/s$ and acceleration within $[-5, 5] m/s^2$, and time was discretized into steps of $0.1s$ for the search. The search heuristic is taken as the minimum remaining time to



(a) (b) (c)

Fig. 4: (a) A complex urban driving scenario involving pedestrians and bicyclists. The car position, pedestrian and bicycle positions, and their paths are overlaid on the map. (b) PT obstacles of the stage 1 problem leading to the first stop sign, with the time-optimal trajectory shown in red. (c) PT obstacles of the stage 2 problem leading to the second stop sign.

the goal position in the absence of obstacles. In the worst-case, the number of nodes A^* generates is exponential in the length of the trajectory, making it unsuitable for problems with long time horizons. On the scenario in Fig. 4a it fails to terminate after more than a minute.

V. APPLICATIONS

Autonomous Vehicles. We demonstrate the ability of our planner to handle the complex scenarios of Fig. 4[1]. We model a car traveling along on Kirkwood Avenue in Bloomington, Indiana on stretch of road with many restaurants and pubs. Bicyclists often share the road lane and pedestrian traffic is heavy, both at and away from crosswalks. The problem is decomposed into two stages: 1) reaching the first stop sign, then 2) reaching a second stop sign. Acceleration bounds are $[-10, 8] m/s^2$ and velocity bounds are $[0, 13.4] m/s$.

Fig. 4b shows the stage 1 trajectory. The car must avoid a bicycle (obstacle 1) moving in front of it. The bicycle accelerates from an initial stop, causing its PT obstacle to be curved initially, and then turns off the road after the stop, so its PT obstacle ends. Near the stop sign the car must avoid a pedestrian (obstacle 2) that cuts in front of the crosswalk.

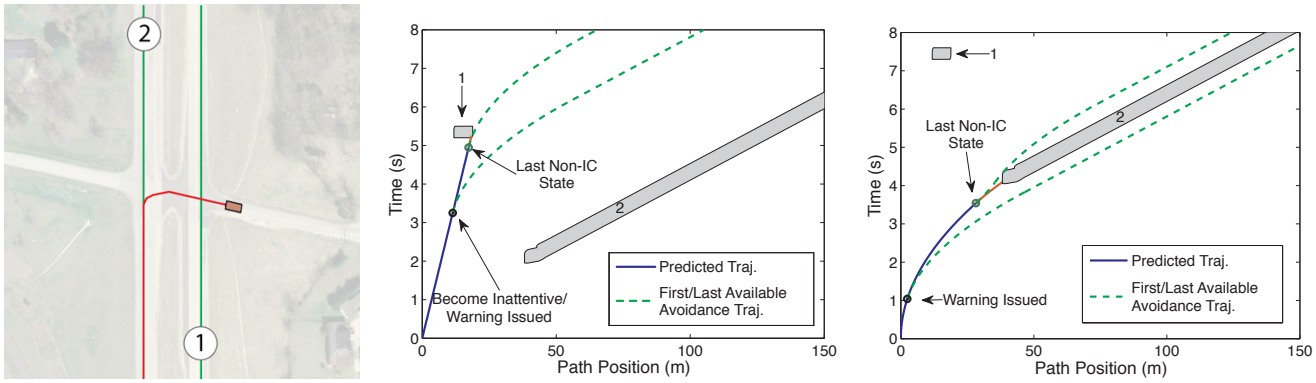
Fig. 4c shows the stage 2 trajectory. The car must now avoid pedestrians in the crosswalk, as well as another bicycle (obstacle 6) that turns into the car's lane. The optimal plan has the car accelerate out in front of the bicycle, then decelerate slightly to avoid a pedestrian (obstacle 7) before going on to the final position. The supplemental video contains simulation of the scenario.

Collision warning systems. Our planner can also be applied to collision warning systems for inattentive drivers. Suppose such a system can detect driver inattention and has a reaction time parameter t_r sufficient for a driver to perceive and respond to a warning, but not so long as to generate unnecessary false positives. Our planner can be called repeatedly to verify that a feasible trajectory exists, assuming the driver continues his/her current behavior up to time t_r. If not, then the driver is about to enter an *inevitable collision state* (ICS), and a warning is issued.

In order to do so, we first collision check the driver's predicted trajectory T_p up to time t_r. If a collision is found, a warning is issued. Otherwise, the planner attempts to find a feasible trajectory starting from the final state of T_p. If none is found, a warning is issued.

Consider a rural highway intersection scenario. The driver attempts to merge south onto State Road 37 in Indiana (Fig. 5a) after crossing two northbound lanes of traffic. The driver incorrectly judges the speed of a northbound vehicle and begins the merge too slowly to cross safely. The planner detects an ICS within t_r and a warning is issued to the driver at the point marked in Fig. 5b. With the appropriate indicators the driver would hopefully be able to accelerate out of the way of the vehicle, or an automated system might take over and guide the car to safety. In a second example with different initial conditions (Fig. 5c), the driver incorrectly judges the speed of the southbound vehicle and

[1]Imagery ©2012 DigitalGlobe, GeoEye, IndianaMap Framework Data, USDA Farm Service Agency



(a) (b) (c)

Fig. 5: Illustration of a collision warning application. (a) An inattentive driver (rectangle) is merging onto the southbound lane of a rural highway, but fails to notice oncoming vehicles (numbered). (b) Given a lead time $t_r = 2.5s$, an inevitable collision state is detected within t_r after the driver become inattentive and a collision warning is issued. (c) An inattentive driver accelerates too slowly while merging and a warning is issued at t_r from the first IC state.

attempts to merge too slowly. The warning indicates that the driver should either slow down or stop at the median, or accelerate ahead of the oncoming vehicle. The supplemental video contains simulation of the scenario.

VI. CONCLUSION

We presented a complete, optimal longitudinal control planner in the presence of moving obstacles. We demonstrated that it can plan time-optimal velocity profiles in cluttered scenarios and be used to detect inevitable collision states in collision warning systems. Simulation tests suggest that the planning system is fast enough for real-time navigation among many dynamic obstacles.

It is possible to further improve the speed of our planner, e.g., using efficient geometric data structures and parallelization to speed collision checking. We also hope to relax some of the assumptions behind our planner, such as allowing it to choose routes along a network of possible paths, and handling more realistic vehicle dynamic models and obstacle behavior models. Eventually, we intend to test our algorithms in more realistic driving simulators and/or real vehicles to better understand how they can be used to improve driving safety.

The supplemental video for this paper is available at:
`http://homes.soic.indiana.edu/jj56/ICRAws.mp4`

REFERENCES

[1] "Roundabout: An informational guide," 2000, in U.S. Department of Transportation Federal Highway Administration Publication Number: FHWA-RD-00-067.

[2] T. C. Ng, J. Ibanez-Guzman, J. Shen, Z. Gong, H. Wang, and C. Cheng, "Vehicle following with obstacle avoidance capabilities in natural environments," in *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, vol. 5, april-1 may 2004, pp. 4283 – 4288 Vol.5.

[3] A. Uno, T. Sakaguchi, and S. Tsugawa, "A merging control algorithm based on inter-vehicle communication," in *Proc. IEEE/IEEJ/JSAI Int. Conf. on Intelligent Transportation Systems*, 1999, pp. 783–787.

[4] J. Johnson and K. Hauser, "Optimal acceleration-bounded trajectory planning in dynamic environments along a specified path," in *International Conference on Robotics and Automation (ICRA)*, St. Paul, USA, May 2012.

[5] J. Reif and M. Sharir, "Motion planning in the presence of moving obstacles," *J. ACM*, vol. 41, no. 4, pp. 764–790, Jul. 1994.

[6] R. Kindel, D. Hsu, J.-C. Latombe, and S. Rock, "Kinodynamic motion planning amidst moving obstacles," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 1, 2000, pp. 537–543.

[7] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[8] J. Leonard, J. How, S. Teller, M. Berger, S. Campbell, G. Fiore, L. Fletcher, E. Frazzoli, A. Huang, S. Karaman, O. Koch, Y. Kuwata, D. Moore, E. Olson, S. Peters, J. Teo, R. Truax, M. Walter, D. Barrett, A. Epstein, K. Maheloni, K. Moyer, T. Jones, R. Buckley, M. Antone, R. Galejs, S. Krishnamurthy, and J. Williams, "A perception-driven autonomous urban vehicle," *J. Field Robot.*, vol. 25, no. 10, pp. 727–774, Oct. 2008.

[9] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[10] S. Kolski, *Mobile Robots: Perception & Navigation*. Pro Literatur Verlag, 2007.

[11] T. Besselmann and M. Morari, "Hybrid Parameter-Varying MPC for Autonomous Vehicle Steering," *European Journal of Control*, vol. 14, no. 5, pp. 418 – 431, 2008.

[12] P. Falcone, F. Borrelli, H. Tseng, J. Asgari, and D. Hrovat, "A hierarchical model predictive control framework for autonomous ground vehicles," in *American Control Conference, 2008*, june 2008, pp. 3719 –3724.

[13] C. Urmson, J. Anhalt, J. A. D. Bagnell, C. R. Baker , R. E. Bittner, J. M. Dolan, D. Duggins, D. Ferguson, T. Galatali, H. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. Howard, A. Kelly, D. Kohanbash, M. Likhachev, N. Miller, K. Peterson, R. Rajkumar, P. Rybski, B. Salesky, S. Scherer, Y.-W. Seo, R. Simmons, S. Singh, J. M. Snider, A. T. Stentz, W. R. L. Whittaker, and J. Ziglar, "Tartan racing: A multi-modal approach to the darpa urban challenge," Robotics Institute, http://archive.darpa.mil/grandchallenge/, Tech. Rep. CMU-RI-TR-, April 2007.

[14] K. Kant and S. W. Zucker, "Toward efficient trajectory planning: the path-velocity decomposition," *Int. J. Rob. Res.*, vol. 5, pp. 72–89, September 1986.

[15] T. Fraichard, "Dynamic trajectory planning with dynamic constraints: A 'state-time space' approach," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 2, jul 1993, pp. 1393–1400.

[16] C. Ó'Dúnlaing, "Motion planning with inertial constraints," *Algorithmica*, vol. 2, no. 1–4, pp. 431–475, 1987.