Combining Task and Motion Planning for Intersection Assistance Systems

Chao Chen[1] and Markus Rickert[1] and Alois Knoll[2]

***Abstract*— A hybrid planning approach is developed for intersection assistance systems up to fully automated driving through intersections. Route planning, task planning and motion planning methods are integrated in a hierarchical planning framework to deal with the various information and constraints in different layers. The navigation agent provides a global driving direction at an intersection according to the selected route. The task planner decides a sequence of actions to accomplish the driving mission taking into consideration traffic rules and semantic conditions. The motion planner generates detailed trajectories to execute the tasks. Meanwhile, the task sequence and the motion trajectory are verified periodically against the actual traffic situation, and re-planning is triggered when necessary in the motion planning or task planning level.**

The hierarchical planning framework is evaluated in several intersection scenarios. The result shows that it can handle the complex planning problems with dynamic objects and provide a modular solution for automated driving that can be easily extended for different traffic rules and applications.

I. Introduction

Intersections and road junctions are the most vulnerable locations for traffic accidents. According to the data from the Federal Statistical Office of Germany [1], about 47.5% of the traffic accidents in Germany in 2013 occurred at intersections or junctions. The statistics from the US [2] show that 40% of the traffic accidents in the United States in 2008 occurred at intersections of which 84.9% caused by incorrect judgement or decision-making of the driver. Therefore, an advanced driver assistance system, which helps the driver to process traffic information and make the right decision at intersections, or drives the vehicle autonomously through them, could greatly improve traffic safety and efficiency.

A typical intersection scenario is illustrated in Fig. 1. The ego vehicle is going to turn left at the intersection with traffic lights, oncoming traffic and pedestrians. In order to make the right decision, different types of information should be evaluated. The information can be obtained with the latest sensor technology. For example, the vehicle location is available from the navigation system to activate the assistance function when an intersection is approaching. The traffic signal recognition, vehicle and pedestrian detection and blind spot surveillance provide a live environment model at the intersection. The information can also come from communication. The infrastructure can maintain a global view of the intersection traffic through external sensors, and inform the relevant vehicles via car-to-infrastructure communication. Vehicles can talk to each other through short-range networks to share information or settle conflicts. These powerful subsystems provide the essential inputs to realize an intersection assistance system.

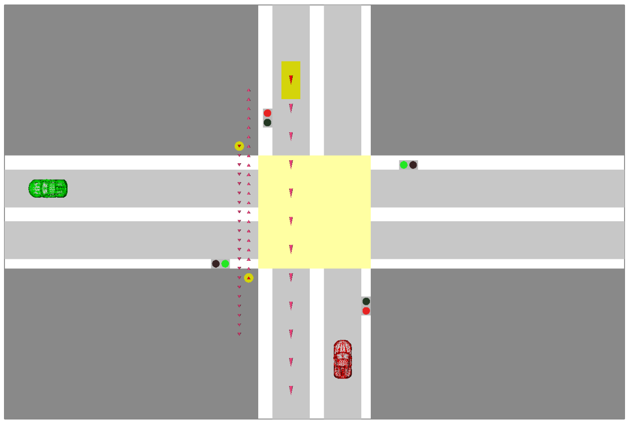


Fig. 1. An sample intersection scenario: an intersection (*light yellow*) with two adjacent lanes (*light grey*) in each direction. The ego vehicle is going to turn left from the start position (*red vehicle*) to the goal position (*green vehicle*). Two pedestrians (*yellow circles*) and another vehicle (*yellow rectangle*) are crossing the intersection at the same time. The lines of small cones indicate their direction. Traffic light signals control the traffic.

The information and knowledge gained from the subsystems is heterogeneous. While some is discrete and symbolic, such as the topology of the roads, traffic lights and signs, some is continuous and concrete, such as the position and speed of objects, geometric shape of traffic lanes and junctions. This paper proposes a hierarchical planning architecture that processes the inputs in three levels: a route planning layer decides the driving direction at an intersection in regards to the road network; a task planning layer refines the driving maneuver to a sequence of tasks concerning traffic signals and rules; and a motion planning layer produces trajectories for each task considering the vehicle kinematics and the obstacle constraints. These three types of planning agents are integrated in a Motion Planning Engine, which enables real-time planning, verification and re-planning to guarantee adaptability and safety of the driving behavior.

II. Related Work

In addition to guiding navigation, the database of a road network [3], [4], [5] can contain information such as traffic signs, traffic lights, and attributes of the traffic lanes. In this case, an autonomous driving agent can reason about not only the topology, but also the priorities of the lane connections in an intersection. A smart intersection can be equipped with

[1]Chao Chen and Markus Rickert are with fortiss GmbH, An-Institut Technische Universität München, Munich, Germany

[2]Alois Knoll is with Robotics and Embedded Systems, Technische Universität München, Munich, Germany

different types of well-positioned sensors to extend the range of the vehicle on-board sensors [6], [7], [8]. The detected vehicles and pedestrians can be mapped to the lanes and connections as a dynamic environment model.

Several approaches have been developed to improve the traffic at intersections using this traffic information. A distributed solution is introduced in [9], which realizes collision avoidance at intersections using a semaphore-based algorithm. Autonomous Intersection Management (AIM) is a multi-agent approach presented in [10], which models and solves the intersection traffic as a grid reservation problem. In this case, an intersection for fully automated vehicles is designed without any traffic lights or stop signs. The Advanced Traffic Management Systems (ATMS) from [11] relies on the communication between vehicles and infrastructure for a coordinated driving behavior at the intersection. The Spatio-Temporal Intersection Protocols (STIP) [12] suggested that several vehicles can negotiate through the vehicle-to-vehicle communication to avoid collisions at intersections. Such a system requires all traffic partners to be sufficient smart with a reliable short range communication.

In comparison to the collaboration solutions, the intersection assistant from [13] employs only the on-board sensors to acquire live information, i.e., omnidirectional cameras for a panoramic view and active cameras for inspection of smaller areas. The prior knowledge of traffic signals and intersection geometry is obtained via a digital map and GPS. In the DARPA Urban Challenge, intelligent vehicles must handle the intersections by themselves. For example, the BOSS [14] has a behavior-based planning architecture to assess the precedence at intersections and plan a yield maneuver when necessary. Such approaches are preferable in the near future as they can function at normal intersections with common traffic members in the absence of C2X communication or smart infrastructure support.

The Space Exploration Guided Heuristic Search [15] is a search-based motion planning method for nonholonomic vehicles. It can be combined with driving task planning [16] in a hybrid planning architecture that deals with symbolic planning problems and motion planning problems at different levels. The contribution of this paper is the integration of a hierarchical planning system in a Motion Planning Engine [17] to achieve real-time planning for an intersection assistance system.

III. System Architecture

The hybrid planning system is illustrated in Fig. 2. The planning is conducted at three layers in such a way that each level extracts knowledge from the according inputs to guide the planning in the next level. A lower level can feedback to a higher level in order to adapt the plan for local changes. The route planning is a general navigation function which determines the waypoints in a road network for a driving destination. The waypoints are passed over to task planning as the milestones. The task planner forward propagates the vehicle states with defined tasks to reach the next milestone. The result of task planning is a sequence of driving tasks, each of which serves as a problem definition for the motion planner, which selects a suitable algorithm to plan the motion for each task. Thus, the system can produce complex automated driving behaviors with a decision process similar to a human driver: The route planning is the long-term strategy related to the route to be taken. The task planning is the short-term decision such as overtaking or lane switching. The motion planning produces the real-time control commands to perform the driving maneuver.

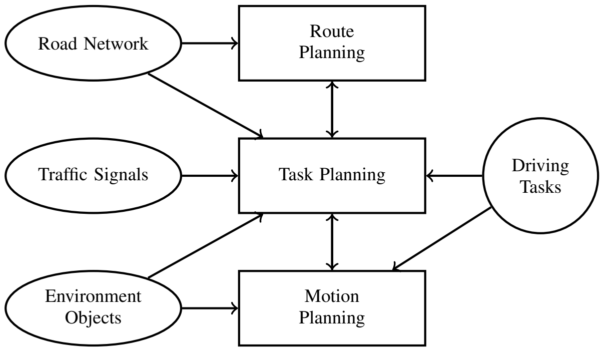


Fig. 2. Hybrid planning system architecture: The modules (rectangles) in the middle are the intelligent agents that do the planning at different levels. The components (ellipses) on the left are the perception inputs that provide real-time information for the planning. The domain specific knowledge (circle) is encapsulated as the driving tasks definition on the right.

The planners in different layers work in different time frames as they solve problems in different scopes. The route planner reacts in seconds; the task planner updates tasks every second; the motion planner produces the trajectory in $100\,\mathrm{ms}$ cycles. Motion Planning Engine runs each planning procedure in two modes: planning and verifying as in Fig. 3. If the initial planning is successful, the engine starts verifying the result repeatedly until the goal is reached. If the result is invalid due to changes in the environment, re-planning is activated to update the solution. A fallback solution is always available so that the vehicle can be brought to a safe state when no solution is found.

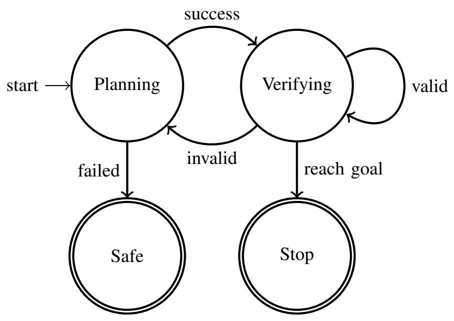


Fig. 3. Motion Planning Engine state diagram

In the prototype implementation, the navigation is simulated with a dummy module. The driving task planning is introduced in [16]. The motion planning is the Space-Time Exploration Guided Heuristic Search method in [18]. Both approaches are extended for the intersection assistance system with details provided in the following sections.

IV. Task Planning

According to the domain definition in [16], the world of driving tasks is modelled with places and objects. The places are *Lane* and *Junction*. An intersection is defined as a junction with the adjacent lanes. The objects are *Vehicle* and *Traffic Signal*. A traffic light is defined as a traffic signal with two states: green and red. The planning agent is assumed to know the state and timing of the light. A traffic light is located at the end of the inbound lanes and mapped to the corresponding connections in the junctions. A new object type *Pedestrian* is introduced to model the people crossing the road. A pedestrian is placed at the beginning or end of the lanes with a time window to cross the road. Thus, the task planner can reason about whether a lane is free of pedestrians in a certain time slot.

The basic driving tasks are defined in [16], such as *FollowLane*, which requires the lane segment to be free in a certain time duration. Now, the free condition should also contain the predicate that no pedestrian is crossing the road during that period. The *ChangeLane* task is the simplest one to drive through an intersection when the given connection is clear and the lanes are free. If a traffic light exists, a green signal is required to perform this task. Furthermore, no traffic from a prioritized connection can occupy the intersection. However, in some cases, the ego vehicle needs to proceed half way to the middle of an intersection and wait for the coming traffic to pass. If there are pedestrians crossing the road, the vehicle should also wait until the target lane is free. Otherwise it may be stopped in the middle of the oncoming traffic and block the intersection. Therefore, two additional tasks: *PrepareChangeLane* and *ProceedChangeLane* are defined in order to divide a change lane maneuver into two parts. Furthermore, a *Wait* task is defined to pause the vehicle for a given time duration when it stops. Details of the three new tasks are as follows.

1) $PrepareChangeLane(V, L_1, L_2, J)$: Vehicle V prepares changing from lane L_1 to lane L_2 at junction J.
Preconditions: Vehicle V should be close to the end of lane L_1 within a range d_{change}. Lane L_1 and L_2 are connected in junction J. The end of lane L_1 should be free during the task duration t_{change}. l_1 is the length of lane L_1. t_0 is the start time. The final time is $t_c = t_0 + t_{\text{change}}$.

$$In(V, L_1, l_1 - d_{\text{change}}, l_1) \wedge Clear(L_1, L_2, J) \\ \wedge Free(L_1, l_1 - d_{\text{change}}, l_1, t_0, t_c)$$

Effects: Vehicle V stops at the middle of connection $C_{1,2}$ from lane L_1 to lane L_2 at time t_c.
Cost: The task duration t_{change}.

2) $ProceedChangeLane(V, L_1, L_2, J)$: Vehicle V proceeds changing from lane L_1 to lane L_2 at junction J.
Preconditions: Vehicle V is at the middle of the connection $C_{1,2}$ from lane L_1 to lane L_2. The begin of lane L_2 should be free during the task duration t_{change}. t_0 is the start time. The final time is $t_c = t_0 + t_{\text{change}}$.

$$At(V, C_{1,2}) \wedge Clear(L_1, L_2, J) \\ \wedge Free(L_2, 0, d_{\text{change}}, t_0, t_c)$$

Effects: Vehicle V is on lane L_2 with distance d_{change} at time t_c.
Cost: The task duration t_{change}.

3) $Wait(V, P, t_{\text{wait}})$: Vehicle V waits at position P for a time duration t_{wait}.
Preconditions: Vehicle V stops at position P. The position P should be free during the task duration t_{wait}. t_0 is the start time. The final time is $t_w = t_0 + t_{\text{wait}}$.

$$At(V, P) \wedge Free(P, t_0, t_w)$$

Effects: Vehicle V stops at position P at time t_w.
Cost: The task duration t_{wait}.

When traffic lights exist, the clear to drive condition in *PrepareChangeLane* only considers the light signal while the one in *ProceedChangeLane* also counts the oncoming traffic from the conflicting lanes. A direct *ChangeLane* task is always preferred as it requires less time for the same maneuver.

The task planning follows a graph search approach. It constructs a graph with nodes as the world states and edges as the tasks changing the states. The time cost of the tasks is used to calculate the actual cost to reach a node, while the heuristic cost reaching the goal state from a node is estimated based on the route in the street network. The planner repeats selecting the node with the least total cost to extend the graph until the goal condition is reached with a smaller actual cost than the total cost of every leaf node. Details of the search algorithm are presented in [16].

V. Motion Planning

The motion planning is performed in the Space Exploration Guided Heuristic Search (SEHS) framework. The SEHS approaches a motion planning problem in two steps. First, it explores the free space with circles to extract the space topology and dimension knowledge. Then, it takes the circle path corridor as heuristics and forward propagates the vehicle states towards the goal. Details can be found in [15], [18]. Experiments show that this approach outperforms the sampling based methods and the grid-based A* search algorithms because its step and resolution adaptation balances the search efficiency and completeness. However, in a practical automated driving scenario, the vehicle motion has additional space and time constraints. The motion is not necessarily permitted in all the physically available space. Neither is all the kinematic possible motion suitable for a driving maneuver. Furthermore, the motion planning and adaptation should happen in real-time. The time requirement is closely related to the constraints. If the scope of the problem and the choice of motion primitives are appropriately selected, the planning algorithm can return a result much faster, especially in negative situations.

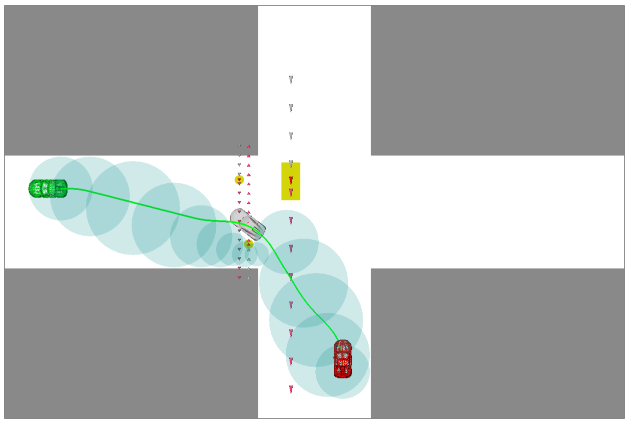
The domain knowledge from task planning can help to decide the context of the motion planning problem of each driving task. First, the start state is decided regarding the current trajectory or the goal state of the previous task. The goal state is selected based on the task, e.g., a *ChangeLane* task ends at the beginning of the target lane, while a *PrepareChangeLane* task suggests a pose in the middle of the intersection. Second, the motion is bounded in a region where the driving maneuver is allowed. For instance, a *ChangeLane* task should be inside the intersection. The planning region can be further shrunk to the exact part of the intersection. Finally, a certain subset of the primitive motions are chosen for the specific task. For example, in a *FollowLane* task, the vehicle should not completely steer away from the lane direction whereas during parking, complex maneuvering is allowed. A *ChangeLane* task provides clues for the steering direction, which is useful to adapt the vehicle motion [17].

In this case, task planning resolves a complex motion planning scenario in a "divide and conquer" manner. It is much more efficient to solve the sub-problem for each task than dealing with the whole motion planning problem. Especially when semantic information is involved, such as traffic rules, it is difficult to handle the logical conditions directly with motion planning. With task planning, however, the domain logic can be tackled at a higher level. Fig. 4 compares the results from the single motion planning and the hybrid approach with task planning. Given all the information about the obstacles, a STEHS algorithm plans a direct motion in Fig. 4(a), which ignores the light signal and the lane borders, and cuts through the intersection for the shortest path. In contrast, the hybrid planning method produces more human-like behaviors waiting for the vehicle and pedestrian to pass, and stays in the lanes. The details of the scenario are explained in Section VI.

As an extended version of SEHS, Space-Time Exploration Guided Heuristic Search is applied to solve the motion planning problem. Because it considers dynamic obstacles and plans in time domain, STEHS is suitable for driving maneuvers with moving vehicles and pedestrians. The Motion Planning Engine in [17] continuously verifies the current motion with the real-time sensing inputs and adapts the trajectory when necessary. Thus, if there is an abrupt change in the environment that cannot be processed by the task planner in time, the motion planning layer can still react in a much shorter time.

VI. Experiments

The hierarchical planning architecture with Motion Planning Engine for the intersection assistance system is evaluated in two scenarios with increasing complexity. The ego vehicle knows the roadmap at the intersection from the navigation module. The traffic signs and traffic lights are detected or communicated to it. It is firstly supposed that the ego vehicle is also aware of the traffic around the intersection, including vehicles and pedestrians. All the algorithms are implemented in C++ and tested on a Linux machine with a 2.9 GHz CPU and 8 GB RAM.


(a) Direct motion planning with STEHS

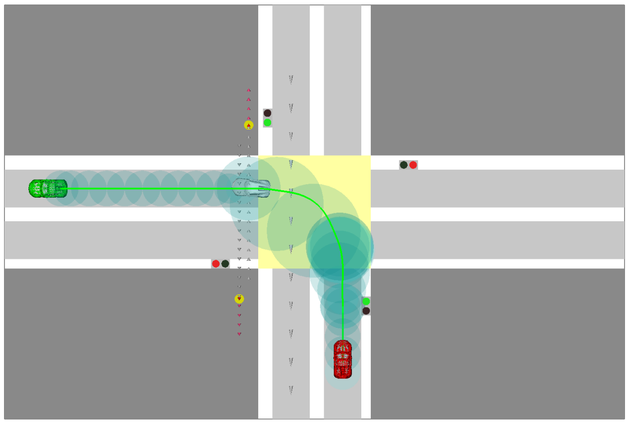

(b) Combining task planning and motion planning

Fig. 4. Comparing direct motion planning and hybrid planning with task planning. The *cyan* circles are the path corridor from space exploration, while the *green* line is the final trajectory from heuristic search.

A. Straight cross with priority to the right

Fig. 5 shows a simple example of driving through an intersection without any traffic lights, signs, or crosswalks for pedestrians. The ego vehicle is driving straight across the intersection while another vehicle is coming from the street to its right. Instead of planning a collision avoidance maneuver as in [18], the ego vehicle should behave adequately regarding the priority to the right rule. According to German driving laws [19], traffic coming from the right has the right-of-way in this situation. Therefore, the ego vehicle should give way at the intersection. Three kinds of tasks are relevant to plan the driving-through-intersection maneuver: *FollowLane*, *ChangeLane*, and *Wait*.

As demonstrated in Fig. 5, four tasks are planned and executed sequentially. First, the vehicle approaches the junction before the stop line with a *FollowLane* task in Fig. 5(a). Then, it stops and waits for the other vehicle to pass through the junction with a *Wait* task in Fig. 5(b). Then, the junction is clear and a motion is planned within the junction area to reach a position at the beginning of the target lane for a *ChangeLane* task in Fig. 5(c). Finally, the vehicle proceeds to the goal position with a *FollowLane* task. The whole process is rather straightforward for the task planning. The

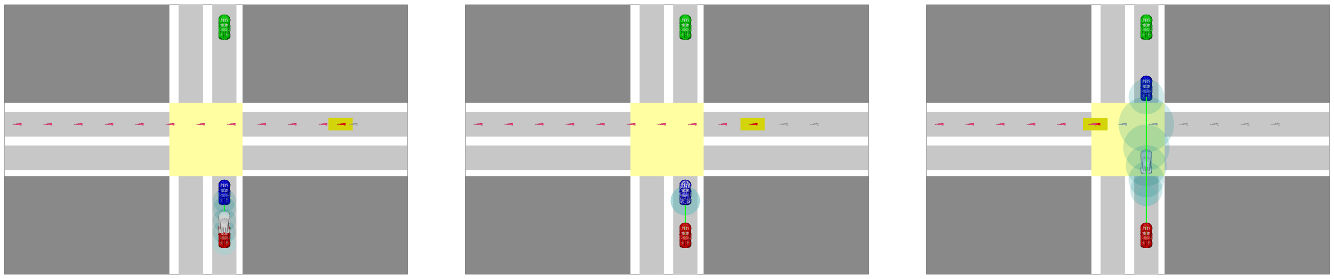

(a) Approach the intersection (b) Wait at the intersection (c) Cross the intersection

Fig. 5. Yielding and driving through an intersection: The *light yellow* area is the intersection. The *light grey* stripes are the adjacent lanes. The *dark grey* is the non-drivable area. The *yellow* rectangle is a vehicle coming from the right. The *red* vehicle shows the start position, while the *green* one the goal position. The internal task goals are in *blue*. The *cyan* circles are the space exploration result. The *green* lines are the motion planning result from the heuristic search.

sub-problems are trivial for the motion planner to solve.

However, the plan above is generated with the assumption that the ego vehicle knows the traffic situation at the intersection in advance. If the planning agent is not aware of the coming vehicle, but perceives the environment in a limited sensing range, the problem is a little more interesting. In this case, the task planner should be aware that anything can happen beyond the sensing range. The vehicle should be able to stop or avoid the collision just in case. Therefore, a better strategy is to adapt the speed of the *FollowLane* task so that the vehicle has enough time to react to the oncoming traffic. In this case, the task planner initially plans a three-step maneuver to cross the intersection without waiting. Later, after the other vehicle is detected, the task planner recognizes that the precondition of the *ChangeLane* task is violated. The tasks are re-planned with a *Wait* task before the *ChangeLane*. The speed of the first *FollowLane* task should be decided depending on the sensing ability and the information availability at the intersection. As a result, the vehicle behaves similar to a human driver who would proceed carefully when the traffic situation is unclear at the intersection.

For such a simple intersection scenario, it is also possible to model driving behavior with a state machine. However, in order to deal with various intersection types, the state machine must have the complexity to handle all possible combinations. Furthermore, if a decision should be made not only based on the current system state but also on the possible evolution of the environment, the state transition becomes much more sophisticated. The advantage of task planning is that the states and conditions are modeled implicitly in the domain definition. With a simple search method, the planner can evaluate all the task combinations for an optimal solution. Moreover, it is easy to extend and modify a subset of tasks for new rules and scenarios without touching the planner. The following experiment is a complicated intersection scenario, which can be solved by extending the task definition used in the current simple example.

B. Left turn with traffic

The example in Fig. 1 is a complex intersection scenario. The ego vehicle is going to turn left at an intersection with traffic lights. Another vehicle is coming from the opposite direction across the junction. Meanwhile, two pedestrians are crossing the road into which the vehicle is planning to turn. The proper driving behavior is turning left when the light is green. In addition, the vehicle should wait half-way into the intersection until the oncoming vehicle and the pedestrians have passed. In this case, the *PrepareChangeLane* and *ProceedChangeLane* tasks come into play.

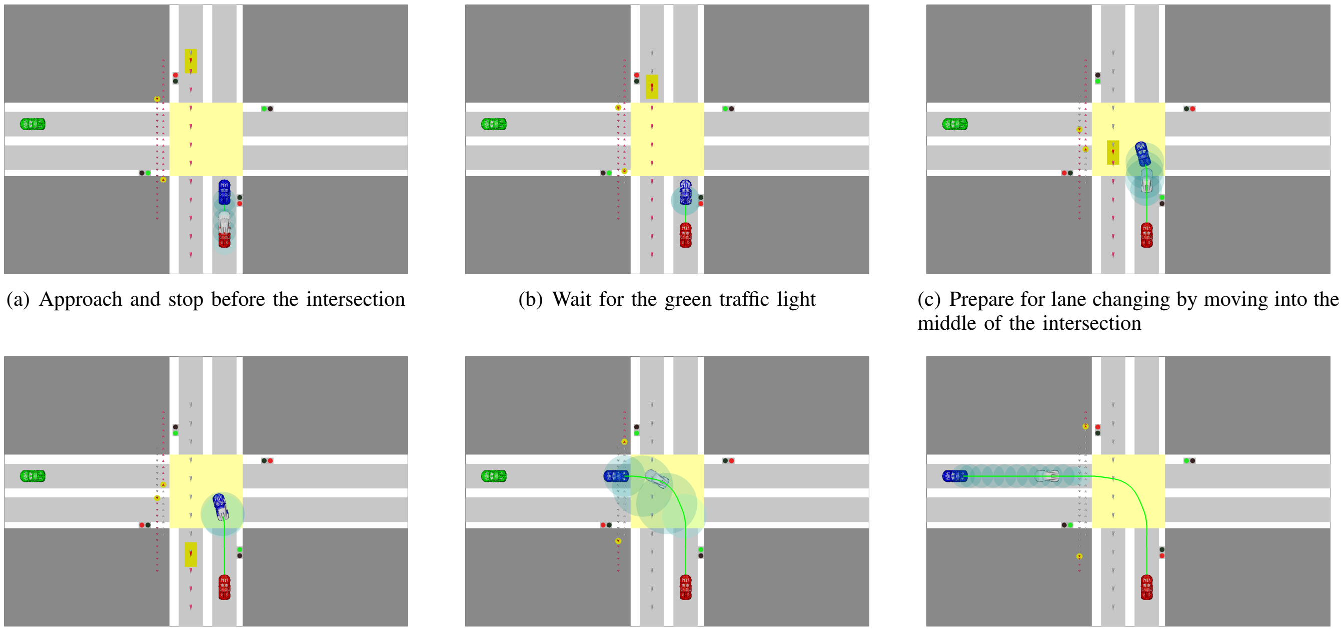
The result is demonstrated in Fig. 6. The ego vehicle first approaches the intersection and stops for the red light in Fig. 6(a) and Fig. 6(b). When the light turns green, the vehicle still cannot directly turn left because the connection is occupied by the oncoming vehicle who has priority. The vehicle can either wait or move to a middle position to prepare for turning. In Fig. 6(c), the vehicle chooses the second option because it saves time by performing the first half of the turning maneuver. After the other vehicle passes by, the ego vehicle remains waiting until the two pedestrians finish crossing the road in Fig. 6(d) and proceeds to turn left in Fig. 6(e). Finally, the vehicle finishes the turning maneuver and goes on driving to the goal in Fig. 6(f).

In this example, the task planner remains unchanged. The domain definition is extended with two tasks. The traffic light and pedestrians are added to the preconditions of the tasks. The task planner then automatically generates a more complicated behavior than in the previous example. Without explicitly coding the situations in a state machine, the planner can decide to perform a left turn in two steps or with a single maneuver. The modified domain definition is backwards compatible, which also works with the previous example. Thus, a knowledge base can be built up in an incremental way for automated driving.

It is important for the task planning that the intentions of other traffic participants can be predicted. The more certain about the environment, the more reliable results can be obtained. However, the task planning can also deal with uncertainties and perception limitations by checking the possible situations or even generating a complete solution.

VII. Conclusion and Future Work

An intersection assistance system is proposed in this work. By combining task planning and motion planning in a hierar-



(a) Approach and stop before the intersection

(b) Wait for the green traffic light

(c) Prepare for lane changing by moving into the middle of the intersection

(d) Wait for the pedestrian to pass

(e) Proceed turning left

(f) Continue driving to the goal

Fig. 6. Turning left at an intersection with traffic lights, oncoming traffic and pedestrians: The *light yellow* area is the intersection. The *light grey* stripes are the adjacent lanes. The *dark grey* is the non-drivable area. The *red* and *green* dots show the traffic light state. The *yellow* rectangle is an oncoming vehicle. The *yellow* circles are two pedestrians crossing the road. The *red* vehicle shows the start position, while the *green* one the goal position. The internal task goals are in *blue*. The *cyan* circles are the space exploration result. The *green* lines are the motion planning result from the heuristic search.

chical planning architecture, the system can handle semantic information and motion planning problems in different layers. Thus, a complex intersection scenario can be divided into sequential tasks, each of which derives a moderate motion planning problem that can be efficiently solved by the STEHS algorithm. As future work, this framework is going to be tested in more intersection scenarios with randomly generated traffic or real datasets. Further studies are planned with imperfect prior knowledge or perception errors in order to achieve robustness and safety in all circumstances.

REFERENCES

[1] Statistisches Bundesamt, "Verkehrsunflle - fachserie 8 reihe 7," 2013.

[2] U.S. Department of Transportation, National Highway Traffic Safety Administration, "Crash factors in intersection-related crashes: An on-scene perspective," 2010.

[3] Defense Advanced Research Projects Agency, "Urban challenge - route network definition file (rndf) and mission data file (mdf) formats," 2007.

[4] J. Knaup and K. Homeier, "Roadgraph - graph based environmental modelling and function independent situation analysis for driver assistance systems," in *Proc. IEEE Annual Conference on Intelligent Transportation Systems*, 2010, pp. 428–432.

[5] M. Dupuis and e.a., "Opendrive format specification, rev. 1.3," 2010.

[6] E. Strigel, D. Meissner, and K. Dietmayer, "Vehicle detection and tracking at intersections by fusing multiple camera views," in *Proc. IEEE Intelligent Vehicles Symposium*, June 2013, pp. 882–887.

[7] R. D. Komguem, R. Stanica, M. Tchuente, and F. Valois, "WARIM: Wireless sensor network architecture for a reliable intersection monitoring," in *Proc. IEEE International Conference on Intelligent Transportation Systems*, Oct. 2014, pp. 1226–1231.

[8] M. S. Shirazi and B. Morris, "Observing behaviors at intersections: A review of recent studies & developments," in *Proc. IEEE Intelligent Vehicles Symposium*, June 2015, pp. 1258–1263.

[9] R. Naumann, R. Rasche, J. Tacken, and C. Tahedi, "Validation and simulation of a decentralized intersection collision avoidance algorithm," in *Proc. IEEE Conference on Intelligent Transportation System*, 1997, pp. 818–823.

[10] K. Dresner and P. Stone, "A multiagent approach to autonomous intersection management," *Journal of Artificial Intelligence Research*, vol. 31, pp. 591–656, 2008.

[11] Q. Jin, G. Wu, K. Boriboonsomsin, and M. Barth, "Advanced intersection management for connected vehicles using a multi-agent systems approach," in *Proc. IEEE Intelligent Vehicles Symposium*, 2012, pp. 932–937.

[12] R. Azimi, G. Bhatia, R. Rajkumar, and P. Mudalige, "Stip: Spatio-temporal intersection protocols for autonomous vehicles," in *Proc. ACM/IEEE International Conference on Cyber-Physical Systems*, 2014, pp. 1–12.

[13] S. Gehrig, S. Wagner, and U. Franke, "System architecture for an intersection assistant fusing image, map, and gps information," in *Proc. IEEE Intelligent Vehicles Symposium*, 2003, pp. 144–149.

[14] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, *et al.*, "Autonomous driving in urban environments: Boss and the Urban Challenge," *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.

[15] C. Chen, M. Rickert, and A. Knoll, "Combining space exploration and heuristic search in online motion planning for nonholonomic vehicles," in *Proc. IEEE Intelligent Vehicles Symposium*, June 2013, pp. 1307–1312.

[16] C. Chen, A. Gaschler, M. Rickert, and A. Knoll, "Task planning for highly automated driving," in *Proc. IEEE Intelligent Vehicles Symposium*, June 2015, pp. 940–945.

[17] C. Chen, M. Rickert, and A. Knoll, "A traffic knowledge aided vehicle motion planning engine based on space exploration guided heuristic search," in *Proc. IEEE Intelligent Vehicles Symposium*, June 2014, pp. 535–540.

[18] C. Chen, M. Rickert, and A. Knoll, "Kinodynamic motion planning with space-time exploration guided heuristic search for car-like robots in dynamic environments," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2015, pp. 2666–2671.

[19] "Straßenverkehrs-Ordnung," 2013, vom 6. März 2013 (BGBl. I S. 367), die durch Artikel 1 der Verordnung vom 22. Oktober 2014 (BGBl. I S. 1635) geändert worden ist.