Hierarchical Trajectory Planning of an Autonomous Car Based on the Integration of a Sampling and an Optimization Method

Wonteak Lim, *Student Member, IEEE*, Seongjin Lee, *Student Member, IEEE*, Myoungho Sunwoo, *Member, IEEE*, and Kichun Jo, *Member, IEEE*

***Abstract*— This paper presents a hierarchical trajectory planning based on the integration of a sampling and an optimization method for urban autonomous driving. To manage a complex driving environment, the upper behavioral trajectory planner searches the macro-scale trajectory to determine the behavior of an autonomous car by using environment models, such as traffic control device and objects. This planner infers reasonable behavior and provides it to the motion trajectory planner. For planning the behavioral trajectory, the sampling-based approach is used due to its advantage of a free-form cost function for discrete models of the driving environments and simplification of the searching area. The lower motion trajectory planner determines the micro-scale trajectory based on the results of the upper trajectory planning with the environment model. The lower planner strongly considers vehicle dynamics within the planned behavior of the behavioral trajectory. Therefore, the planning space of the lower planner can be limited, allowing for improvement of the efficiency of the numerical optimization of the lower planner to find the best trajectory. For the motion trajectory planning, the numerical optimization is applied due to its advantages of a mathematical model for the continuous elements of the driving environments and low computation to converge minima in the convex function. The proposed algorithms of the sampling-based behavioral and optimization-based motion trajectory were evaluated through experiments in various scenarios of an urban area.**

***Index Terms*— Hierarchical trajectory planning, behavioral trajectory, motion trajectory, autonomous car, trajectory generation.**

Manuscript received January 21, 2017; revised June 21, 2017; accepted September 15, 2017. Date of publication January 1, 2018; date of current version February 1, 2018. This work was supported in part by the BK21 Plus Program under the Ministry of Education, Republic of Korea, under Grant 22A20130000045, in part by the Industrial Strategy Technology Development Program under Grant 10039673, in part by the Industrial Strategy Technology Development Program under Grant 10042633, in part by the System Industrial Strategic Technology Development Program under Grant 10079961, in part by the Industrial Strategic Technology Development Program under Grant 10060068, in part by the International Collaborative Research and Development Program under the Ministry of Trade, Industry and Energy (MOTIE Korea), under Grant N0001992, and in part by the Energy Resource Research and Development Program under the Ministry of Knowledge Economy, Republic of Korea, under Grant 2006ETR11P091C. The Associate Editor for this paper was P. Zingaretti. *(Corresponding author: Kichun Jo.)*

W. Lim, S. Lee, and M. Sunwoo are with the Department of Automotive Engineering, Hanyang University, Seoul 133-791, South Korea.

K. Jo is with Valeo Driving Assistance Research, 93012 Bobigny, France (e-mail: kichun.jo@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2017.2756099

I. Introduction

AUTONOMOUS driving navigates a car from the current position to the target position based on the understanding of driving environments. For safe and comfortable navigation, *trajectory planning* plans the optimal maneuver and trajectory to avoid collisions with obstacles within the physical limits of the vehicle [1], [2]. In particular, *trajectory planning* in urban areas must manage the various driving situations caused by different types of roads, traffic regulations, barriers, and traffic participants (vehicles, bikes, bicycles, and pedestrians). In addition, the states (position and velocity) of the participants change dynamically. To consider the spatiotemporal characteristics of the movement, *trajectory planning* is more sophisticated. In other words, the *trajectory planning* requires an ability handle the complex and various elements of the dynamic urban environments efficiently.

For *trajectory planning*, there are two approaches: sampling-based planning [3]–[7] and numerical optimization-based planning [8], [9]. Sampling-based trajectory planning generates the trajectory by making a variety of trajectory candidates based on predefined patterns and then selects the best one with the minimum cost of safety, comfort, and energy. The advantage of the sampling method is its free-form cost function. Since the various forms of a cost function are available, the trajectory candidates can be designed by continuous mathematical models of vehicle dynamics and road geometry as well as discrete environmental models of the status of the traffic light, the position of the stop line, and types of traffic signs. However, this planning method requires significant computational resources in order to consider all the possible motions of the vehicle because it requires a tremendous number of candidates to represent continuous motion as the discrete samples.

Numerical optimization-based planning finds the trajectory based on a differentiable cost function and constraints. If both the cost function and the constraints are convex functions, the trajectory can converge into the global optimum. In trajectory planning, geometric elements (road curvature and road boundary) and vehicle dynamics elements (velocity, acceleration, yaw rate, and jerk) are represented by differentiable mathematical functions. Thus, these elements can be effectively applied

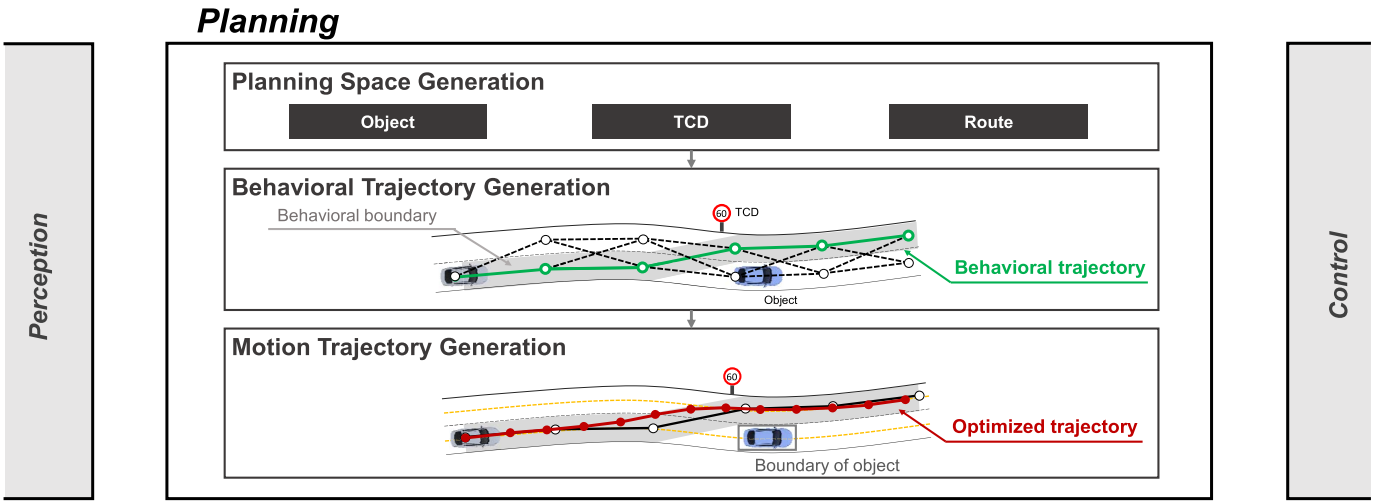

Fig. 1. System architecture of hierarchical trajectory planning.

to numerical optimization-based trajectory planning. On the other hand, the numerical method is not able to consider all of the driving environments of an urban area because some elements including a traffic sign, a traffic light, and the number of lanes are modeled by discrete states. In addition, modeling of complex driving situations by using convex functions is a challenging problem.

As mentioned above, the two types of trajectory planners have different characteristics. The sampling-based planning has a strong ability to deal with discrete elements. In addition, the method can simplify the large problem by the sampling patterns. For example, a lattice-based planner [4], [5], [10] represents the configuration space for planning as a lattice pattern. By using the pattern, the complex planning problem is converted into a simple graph search problem. On the other hand, numerical optimization-based planning is specialized in the continuous mathematical problem. If it is a convex function, the solution can be found in real time [8].

Despite the complementary characteristics of the two methods, previous studies of hierarchical trajectory planning [10]–[12] did not combine the two approaches for the trajectory planning. However, integration of these two methods is likely to be more efficient to manage all the complex urban elements including both the discrete and continuous environmental models. For example, traffic lights and traffic signs have a discrete state in a specific position. Since these elements cannot be represented by a mathematical model, sampling-based trajectory planning is more suitable than the other one. On the other hand, the elements of vehicle dynamics, a road geometry, and driving-allowable boundaries are represented as a mathematical model. Thus, the numerical optimization-based method is better than the sampling-based one.

For the synergy effect of two planning methods, this paper proposes hierarchical trajectory planning based on the integration of a sampling and a numerical optimization technique for urban autonomous driving. The trajectory planner consists of two modules: a *behavioral trajectory planning* module and a *motion trajectory planning* module. The *behavioral trajectory planning* module generates a macroscopic trajectory (or a maneuver trajectory) by using the sampling-based method. This method concentrates on the discrete environmental elements of traffic rules, road types, and surrounding objects as well as the simplified kinematic and dynamic characteristics of the vehicle. Based on the *behavioral trajectory*, *motion trajectory planning* searches for a microscopic trajectory to control the vehicle. This trajectory mainly considers the vehicle dynamics and the collision avoidance with surrounding objects through numerical optimization. The proposed hierarchical trajectory planning was implemented in a test vehicle A1 [28] and the performance was evaluated in various driving scenarios.

This paper is organized as follows. Section II presents the system architecture of the proposed algorithm. Section III-V describes the detailed algorithm: *planning space*, *behavioral trajectory planning*, and *motion trajectory planning*. Section VI evaluates the hierarchical trajectory planner by vehicle tests. Finally, Section V provides the conclusion.

II. System Architecture

To generate the trajectory in an urban road, *trajectory planning* must consider the complex driving environments consisting of surrounding vehicles, pedestrians, road types, road lanes, traffic rules, and traffic signals. Based on understanding the situations, the final trajectory should be determined within the limits of the vehicle dynamics. To consider the various elements effectively, this paper proposed the hierarchical trajectory planning. The overall architecture of this planner is described in Fig. 1. The planner has three steps: *planning space*, *behavioral trajectory planning*, and *motion trajectory planning*.

For the simplification of various driving environments perceived from sensors, this paper defines *planning space*. This space is composed of three components: *object*, traffic control device (*TCD*), and *route*. The *object* means a spatial occupant that has a risk to collide with the ego vehicle. For example,

there are barriers, traffic cones, vehicles, bikes, and pedestrians. The *TCD* contains a traffic light, sign, road marker, and speed bump. The *route* stands for road connection, road geometry, lane information, and lane boundary.

Behavioral trajectory planning determines the maneuver based on the current and predicted driving situations. The situations can be diverse by the number of lanes, the positions, and the velocities of the objects, and the existence and state of TCDs. These elements cannot be modeled by continuous mathematical models. To apply the discrete elements, the sampling-based planning method is used. The sampling space of the method is made by the simplified lane information of the *route*. From the space, the behavioral candidates are extracted according to the predefined patterns. The best of the candidates is selected by considering the objects and TCDs. The final *behavioral trajectory* provides the reference trajectory nodes and the boundary area to guide the future motion of the ego vehicle.

In the reduced searching area of the *behavioral trajectory*, the *motion trajectory* finds a smooth trajectory by using the numerical optimization-based method. This method focuses on the mathematical models of the *route* (a road geometry and road boundary), the *object* (distances of objects), and the vehicle dynamics. Based on the models, a cost function for the numerical optimization is designed by driving comfort and energy. The safe-related elements such as a collision, a driving boundary, and a physical limit of the vehicle are considered by the constraints and the boundary conditions of the optimization problem. With this data, the numerical optimization problem is represented as a convex function. Due to the reduced searching area and the convex property, there is a significant saving in the computation time to optimize the local trajectory.

III. Planning Space

Planning space is a data abstraction layer to represent the surrounding environments for both *behavioral trajectory planning* and *motion trajectory planning*. Through the *planning space*, the trajectory planners can manage various types of environmental information with common interfaces [13]–[15]. This paper designs a *planning space* that consists of three elements: *objects*, *TCD*, and *route*.

A. Object

An *object* is a spatial obstacle to be avoided. The types of objects are divided into static and dynamic objects. Static objects represent stationary objects that have no velocity. This type of object can be represented by a point list or various types of grid maps (occupancy grid map [10], [16] and evidential grid map [17], [18]). This paper uses an occupancy grid map to represent the state of the static objects. Each cell of the occupancy grid map has a probability where the static objects exist or do not. The occupancy grid map can be extracted from the range sensors, such as light detection and ranging (LIDAR) sensors, radio detection and ranging (RADAR) sensors, and stereo cameras.

Dynamic objects are moving traffic participants including the surrounding vehicles, bicycles, and pedestrians. This paper implements dynamic objects by using a bounding box model having the position, heading angle, shape, and size information of the objects [19], [20]. The information is used to predict the future trajectory and assess the risk of collision during trajectory planning.

B. TCD

The *TCD* are the devices to control the traffic flow and inform the traffic regulations to drivers. Examples of the *TCD* are traffic lights, traffic signs, speed bumps, and road markers. These TCD can be used to determine the behavior and plan the trajectory of the ego vehicle. For example, the vehicle will stop and wait in front of a stop line when the traffic light is red. After the light changes to green, the vehicle starts to go through the intersection. The *TCD*s are managed by a combined behavior model [10] or a state machine [9]. This paper uses a state machine to deal with various types of TCDs.

C. Route

The *Route* provides the road information to the trajectory planning to reach the destination. The *route* has a hierarchical structure; it is composed of a *global route* and a *local route*. The *global route* has sequential roads connected from the start position to the destination position. These are extracted from the road network map considering the travel distance, time, and traffic flow [21]. This plays an important role to select a single route that the ego vehicle should follow among the various route candidates due to different road types of an intersection, merge, split, and roundabout road. Regardless of the roads, the global route can generate a single route with sequential roads. Based on the road sequence, the *local route* provides more detailed road information including road geometry, road boundary, and lane information. The road geometry, in particular, is crucial information to understand the shape of a road for the planning. The shape can be modeled by discrete waypoints or many types of mathematical curves (polynomial curve, clothoid [22]–[24], and spline [25]–[27]). This paper uses the B-spline representation [27] for the model of roadway geometry. The road boundary divides the road into two areas: an allowable area for driving or not. The boundary is extracted from lanes or guardrails as the same data structure like the road geometry or driving corridor [9]. In addition, the *local route* is represented on the local coordinate of the ego vehicle and can be updated by a lane detection camera or a precise map in a real-time.

IV. Behavioral Trajectory Planning

Motion planning for autonomous driving consists of behavior planning and local trajectory planning [2]. The behavior planning determines the maneuver in various driving situations. The types of maneuvers include lane keeping, lane change, stop, acceleration, and deceleration. To select the best one, the state-machine is widely used [10], [11], [16], [28]. This method predefines the behaviors as discrete states and then selects the minimum-cost state. The state divides the complex problem of the general trajectory planning into simple

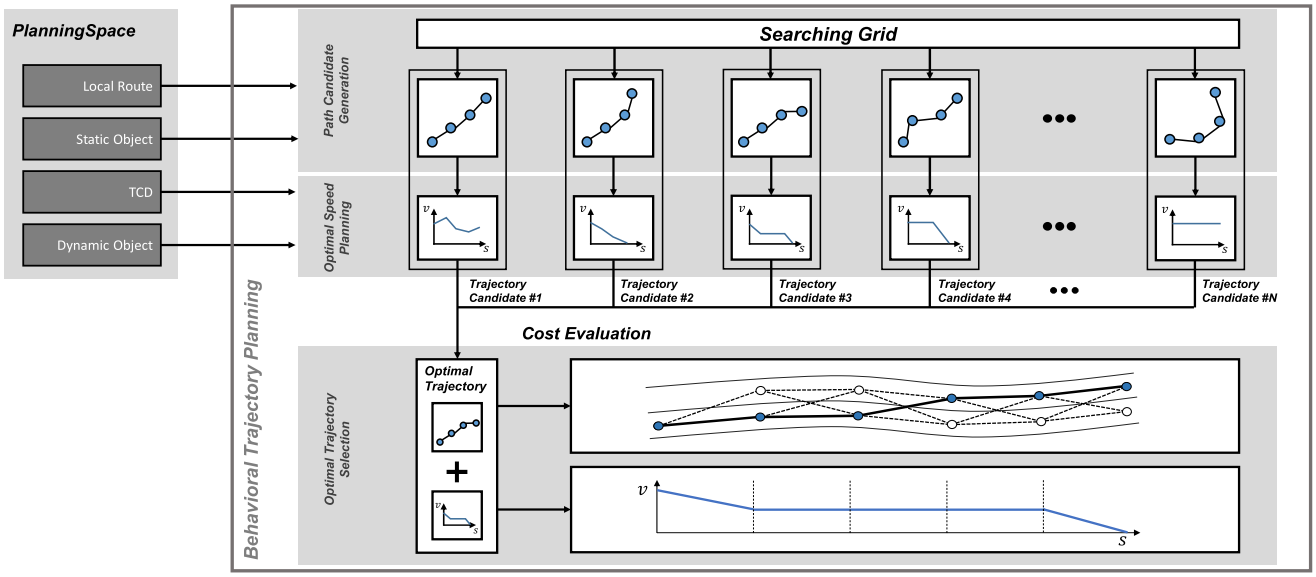

Fig. 2. Overall architecture of behavioral trajectory planning: path candidate generation, optimal speed planning, and optimal trajectory planning.

independent sub-problems according to each maneuver. Based on this separation, the local trajectory planning can only focus on each sub-problem. However, the planning algorithms based on state machines needs many states to manage various maneuvers. It complicates the whole of motion planning algorithm for the general purpose of urban driving and cannot cover all the driving situations because of the countless situations. Furthermore, state-machine cannot express the behavior change over time because just one behavior is selected each time. To solve these problems, this paper proposes *behavioral trajectory planning* as behavior planning

The *Behavioral trajectory* represents the behavior by using the data structure including the sequential nodes of the position and time information. The spatiotemporal nodes are generated by the path-velocity decomposition method [29], [30]. To utilize this method, *behavioral trajectory planning* has three steps: path candidate generation, optimal speed profile generation, and trajectory selection. The overall process of the *behavioral trajectory planning* is described in Fig. 2.

A. Path Candidate Generation

1) Curvilinear Coordinate Conversion: Curvilinear coordinates are a coordinate system which is established following the curve geometry [31]. In the coordinate, there are an axis (s-axis) tangent to the curve and one or two axes (n-axis) orthogonal to the s-axis. Through the curvilinear coordinate conversion, it is simple to explore the points on the curve regardless of the curve geometry because the points are placed on the s-axis of the curvilinear coordinate.

Road geometry in the 2D Cartesian coordinate is converted into one in the 2D curvilinear coordinates as shown in Fig. 3. Through the coordinate conversion, the complex geometric relation between the lanes [Fig. 3(a)] can be converted into a simple one in the orthogonal computing space [Fig. 3(b)]. In the orthogonal space, it is a simple to represent each lane by using the offset of the n-axis. Based on the advantages

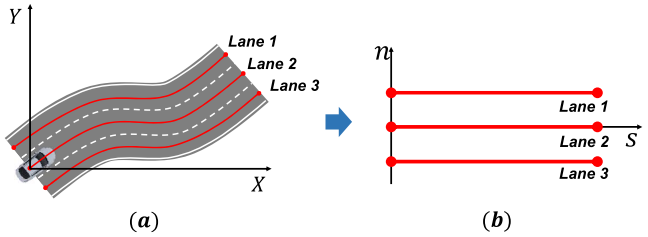

Fig. 3. Curvilinear coordinate of road geometry. (a) Cartesian coordinate. (b) Curvilinear coordinate.

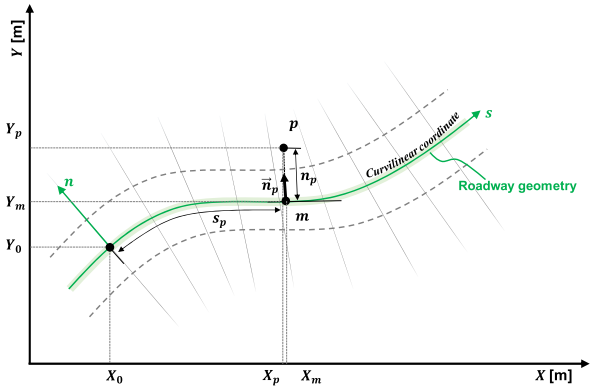

Fig. 4. Curvilinear coordinate conversion between the Cartesian coordinate and a road geometry-based coordinate.

of curvilinear coordinate, the *behavioral trajectory planning* abstracts vehicle motions on various road shapes in the simple orthogonal space.

Two coordinate conversions are required to use curvilinear coordinates for the *behavioral trajectory planning*. The first conversion is from $(s,\ n)_{curvilinear}$ to $(x_p,\ y_p)_{cartesian}$, as shown in Fig. 4. This conversion searches for the point $(x_m,\ y_m)_{cartesian}$ of which the arc length from $(x_0,\ y_0)_{cartesian}$ is s_p. At the point, $(x_p,\ y_p)_{cartesian}$ is calculated by using the perpendicular vector, $\vec{n}_p$ and the perpendicular distance, n_p. The second conversion from $(x_p,\ y_p)_{cartesian}$ to

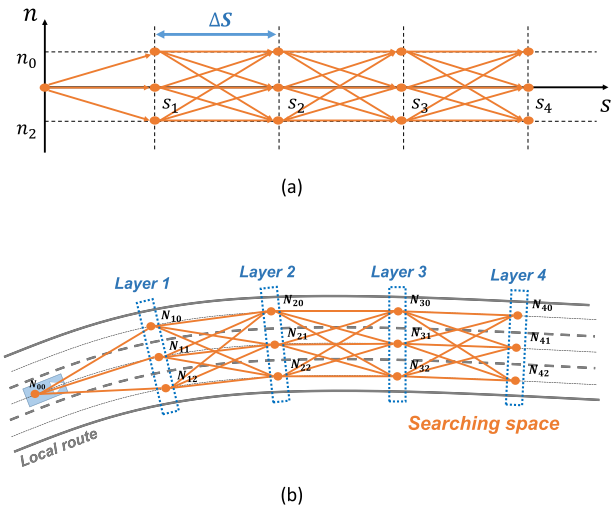

Fig. 5. Searching space can simplify the trajectory planning problem by using a simple pattern. (a) Searching space in curvilinear coordinate. (b) Searching space in cartesian coordinate.

$(s, n)_{curvilinear}$ finds the point $(x_m, y_m)_{cartesian}$ on the curve by the closet point method [32]. This method uses a quadratic minimization and Newtown's method to calculate the minimum distance between the point and the geometry curve.

2) Searching Space Generation: Searching space is the basic grid to represent various behavioral candidates on the *local route*. The grid is configured by a directed graph structure in the curvilinear coordinate, as shown in Fig. 5(a). The grid spacing is determined by vertices N_{ij} and Δs. Each vertex N_{ij} is generated at each lane from the position of the ego vehicle as follows

$$\mathrm{s} = i\,\Delta s, \tag{1}$$

$$n = n_j \tag{2}$$

where i and j are the indices of the s and n-direction, respectively. In the on-road environment, a macro maneuver in a lateral direction can be determined in a lane-scale level. For this reason, the offset n_j refers to the jst lane offset from the ego's lane. The intervals Δs is the unit distance to divide vehicle's maneuvers in a longitudinal direction. It is determined by the speed of the vehicle. However, this interval should be larger than the minimum interval Δs_{min} even if the speed is zero. This minimum value should be configured considering the curvature constraint k_{limit} which is determined by the minimum turning radius of the vehicle. The sets of vertices at the same s are defined as the *layer* in Fig. 5(b). Through the combination of *layer*s, the edges of the searching space are constructed.

3) Path Candidate Extraction From the Searching Space: Behavioral path candidates are a set of sequential edges from the searching space graph. These candidates are generated by two steps. The first step is to prune the edges that are not drivable due to the static objects. To check the collision between the edges and the static objects, an occupancy grid map is used [33]. The edges overlapped by static objects mean that the collision must occur regardless of the time. For this reason, the edges are removed from searching space graph.

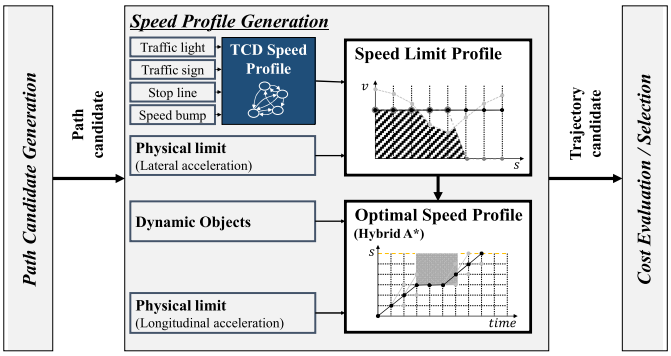

Fig. 6. Process of speed profile generation that consists speed limit profile and optimal speed profile generation.

The second step is to extract the candidates from the pruned searching space graph. All the set of the consecutive edges from $layer_0$ to $layer_N$ are converted into behavioral path candidates.

B. Speed Profile Generation

Behavioral trajectory must have spatiotemporal information. However, behavioral path candidates contain only spatial nodes. To add the temporal information to each candidate, *speed profile generation* is used. The optimal speed profile considers *TCD*, physical limit, and dynamic objects, as shown in Fig. 6. For efficient speed generation, it has two steps: the speed limit profile generation and the optimal speed search.

Speed limit profile refers to the maximum speed limits along the s axis of the road geometry. These are determined by the maximum lateral acceleration and *TCD*. The maximum lateral acceleration should be selected considering the physical limit of the vehicle and the driving comfort. Based on the maximum acceleration a_{lat} and the road curvature k, the speed limit v_{acc} can be calculated by

$$v_{acc} = \sqrt{\frac{a_{lat}}{k}}. \tag{3}$$

TCD speed profile is a module that provides the regular speed of *TCD*s for the speed limit profile. In urban environments, there are various types of *TCD*s. In addition, each TCD has various states to give a warning and signal information to the drivers. To manage many types and states of *TCD*s, it is the best choice to use a state machine. The types of *TCD* are configured as the *state*s of the state machine. The outputs of the state machine contain the speed limit and the position (or the effective region which is the road area that affects TCD). For example, a state of a red traffic light generates the speed limit (0 km/h) and the effective region from the corresponding stop line.

Speed profile generation finds the best speed profile based on the speed limit and dynamic objects on the Space-Time domain (*S-T* domain) [34], [35]. Time and space axis means predictive time and position along the behavioral path respectively. In Fig. 7, the point p_0 represents that the arc length from the current position at time t_0 is the space s_0. In the *S-T* domain, a dynamic object can be drawn from the gray region

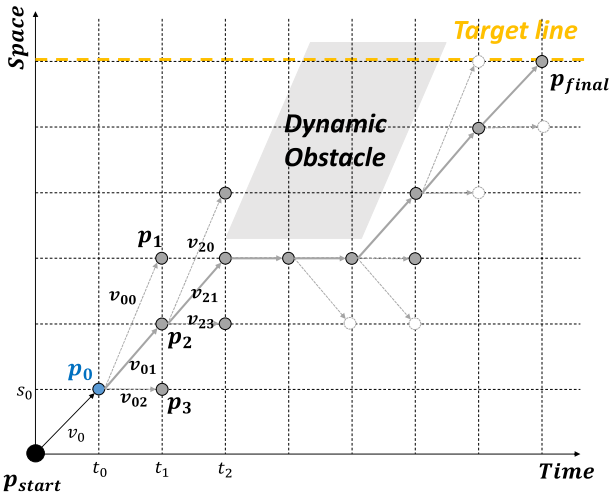

Fig. 7. Hybrid A* for the optimal speed generation considering the speed limits and the acceleration limits.

in Fig. 7. To avoid collision with dynamic objects, the speed profile should be generated in the white region excluding the gray region.

To solve the problem, the Hybrid A∗ algorithm [36] is used in the S-T domain. The first step is to spread the candidate speeds at point P_i. The speed candidate $v_{i,j}$ is generated within the maximum longitudinal acceleration a_{maxlon} and the speed limit generated above.

$$\frac{v_{i,k} - v_{\mathrm{i-1,j}}}{\Delta t} \leq \mathrm{a}_{maxlon}, \tag{4}$$

$$v_{i,k} \leq \mathrm{v}_{limit}(s_i) \tag{5}$$

where i is the point index and j, k are the speed candidate index.

The second step is to apply the performance index f_{speed} of each speed candidate. The indices are calculated by the weighted sum of the reference cost f_r, acceleration cost f_a, jerk cost f_j, and consistency cost f_c as follows:

$$f_{speed} = w_r f_r + w_a f_a + w_j f_j + w_c f_c \tag{6}$$

where w_r, w_a, w_j and w_c are the weights of the reference, acceleration, jerk, and consistency.

The reference cost f_r is defined as

$$f_r = |v_{limit}(s_i) - v_{ik}| \tag{7}$$

where v_{limit} is the speed limit from the *speed limit profile* module and i and j are the indices of the path candidate node and the speed candidate at the node, respectively. The cost reduces the travel time to the destination. The acceleration and jerk cost is related to driving comfort.

Each term is described as

$$f_a = \frac{\left|v_{i,k} - v_{\mathrm{i-1,j}}\right|}{\Delta t}, \tag{8}$$

$$f_j = \frac{\left|v_{i,k} - 2v_{\mathrm{i-1,j}} + v_{\mathrm{i-2,j}}\right|}{\Delta t^2}. \tag{9}$$

The last term is the consistency cost f_c which is

$$f_c = \frac{\left|v_{t,i,k} - v_{t-1}(s_i)\right|}{\Delta t^2} \tag{10}$$

where index t refers to the time index. This cost reduces the rapid change of the speed profiles determined at the previous time and the current time. The smooth transition can enhance the driving comfort.

The third step is to check the collision with the dynamic object and the list of the speed candidates. At point P_i, the temporal trajectory in the *S-T* domain can be estimated based on the constant speed model during step time Δt. If the speed candidate is collision-free, it is added to the *open list* of A∗.

The final step is to pop up the optimal speed candidate of *open list* and to add it to the *closed list*. The five steps are repeated until the speed candidate in the *closed list* arrives at the target line. The final speed profile is constructed by using the reverse direction search in the *closed list* [36].

C. Optimal Trajectory Selection

The behavioral trajectory candidate represents various maneuvers for autonomous driving. Among these maneuvers, the best one is determined by the performance index:

$$G(i) = w_{LC}C_{LC}(i) + w_{dis}C_{dis}(i) + w_{cons}C_{cons}(i) \tag{11}$$

where i indicates the index of the behavioral trajectory candidates.

The lane change cost C_{LC} means the number of lane changes of the trajectory candidate. An unnecessary lane change is the dominant factor to consume the energy and reduces driving comfort. To protect these, the cost is calculated by

$$C_{LC} = \sum_{k=1}^{N_{path}-1} |Lane(n_k) - Lane(n_{k-1})| \tag{12}$$

where $Lane(n_k)$ refers to the lane index of n_k which is the node index of the path candidate, and N_{path} is the number of the path candidate nodes.

The travel distance cost C_{dis} indicates the travel distance for the prediction time $T_{predict}$. Since trajectory candidates have different speed profiles, the travel distances during the prediction time are various. Among these distances, drivers tend to choose the shortest one because the shorter the trajectory, the greater the time saved. The distance is calculated by the integration of the speed profile:

$$C_{dis} = v_{regular}T_{predict} - \int_{t=0}^{T_{predict}} v(t)\,dt. \tag{13}$$

The consistency cost C_{Cons} refers to the continuous trajectory generation. Trajectory planning determines the behavior every execution. Since the driving environment is changed for the transition time, a behavior trajectory different from the previous one can be selected every execution. However, frequent changes of the behavior decrease the driving quality and control performance. To solve this problem, the consistency cost prevents unnecessary behavior change compared to the previous *behavioral trajectory*. To measure the difference, trajectory nodes X_t^j and X_{t-1}^j are resampled from the previous and current *behavioral trajectories* at the sampling time ΔT_{cons}. The total number N of the nodes is determined by

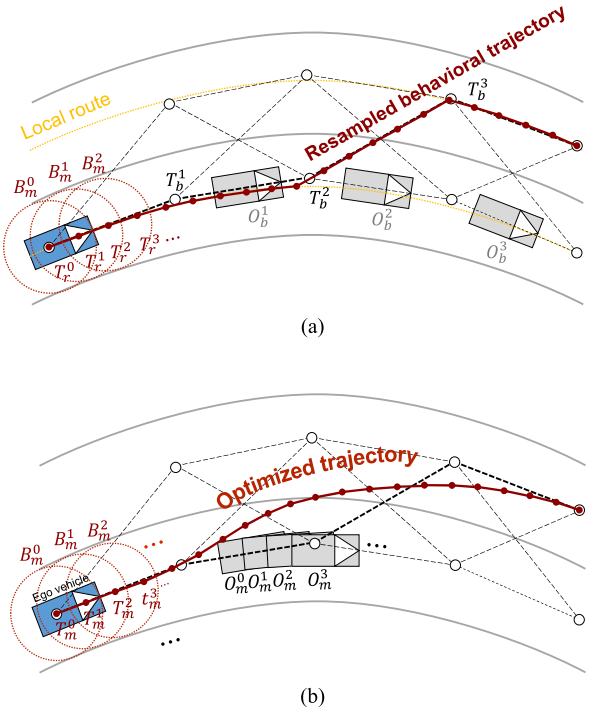

Fig. 8. Motion trajectory planning based on the resampled behavioral trajectory including the reference trajectory $\mathbf{T_r^t}$ and the boundary information $\mathbf{B_r^t}$ where t is the time index and r refers to the resampled behavioral trajectory. (a) Resampled behavioral trajectory. (b) Optimized motion trajectory.

the prediction time $T_{predict}$ and the sampling time ΔT_{cons}. The consistency of the sampled trajectory is measured by

$$C_{cons} = \sum_{j=0}^{N_b-k} \left| X_t^j - X_{t-1}^{j+k} \right| \tag{14}$$

where t refers to the generated time and j indicates the index of the trajectory's node. N_b means the total of the trajectory's nodes where k is the synchronization index which is calculated by the elapsed time $\Delta T_{elapsed}$ and resampling time ΔT_{cons}.

V. Motion Trajectory Planning

The *Behavioral trajectory* through the sampling method determines the maneuver of the vehicle by considering discrete information such as traffic control devices and obstacles. However, the *behavioral trajectory* has a limitation to represent a local trajectory that a vehicle should follow because it does not consider vehicle kinematics or dynamics or reflect the boundary information of the lane and obstacles. Furthermore, the *behavioral trajectory* lacks the resolution to express the continuous change of a vehicle state. To solve the problem, *motion trajectory planning* converts *behavioral trajectory* into a denser trajectory that reflects the vehicle dynamics, road geometry, object boundary information, and *behavioral trajectory* information that contains the reference trajectory and the boundaries of the nodes as shown in Fig. 8. This transformation has two steps: behavioral trajectory resampling and numerical optimization for a navigation-level trajectory. From the behavioral trajectory, the first step extracts a reference trajectory that a motion trajectory should trace. This reference means a guideline to maintain the consistent behavior during motion trajectory planning. Based on the resampled behavioral trajectory, the optimization step generates the final motion trajectory by using a numerical optimization programming.

A. Resampled Behavioral Trajectory

A maneuver that autonomous vehicles should follow is decided by the *behavioral trajectory*. The trajectory is the macro-scale maneuver with a low resolution. For this reason, it is difficult to represent the road geometry since *behavioral trajectory* uses the linear interpolation between two discrete sample nodes. Therefore, this process is necessary to increase the resolution of the *behavioral trajectory* through the resampling step time of *behavioral trajectory* $T_b^0, T_b^1, T_b^2, \cdots$ [black points in Fig. 8(a)]. For example, the optimized trajectory using only *behavioral trajectory* in Fig. 8(a) has difficulty with obstacles while considering vehicle dynamics because of its resolution. To overcome this limitation, we convert the coordinate of the line segment to curvilinear coordinate and split the segment into samples $(T_r^0, T_r^1, T_r^2, \cdots)$ [red points in Fig. 8(b)] with new step time $\Delta T_{resample}$.

B. Numerical Optimization for Motion Trajectory Planning

1) Objective Function: Motion trajectory planning must generate a feasible trajectory considering the vehicle dynamics which is defined in the Cartesian coordinate. To represent it efficiently, the motion trajectory is defined as $\{\mathbf{T}_m^i\} = \{(x_i, y_i)\}$ in the same coordinate, as shown in Fig. 8(b). It should keep the resampled *behavioral trajectory* to follow the planned maneuver and should consider vehicle dynamics for driving comfort. To satisfy the considerations, the objective function f_{motion} consists of three terms:

$$f_{motion} = \sum_{i=0}^{N_m} (f_{ref} + f_{acc} + f_{jerk}). \tag{15}$$

where N_m refers to the total number of *motion trajectory*'s nodes.

The first term f_{ref}, called the reference cost, is defined as

$$f_{ref} = \left((x_{i,ref} - x_i)^2 + (y_{i,ref} - y_i)^2 \right) \tag{16}$$

to follow road geometry of the resampled *behavioral trajectory* where $(x_{i,ref}, y_{i,ref})$ is a node T_r^i of the resampled *behavioral trajectory*, as shown in Fig. 8(b). The value of the cost increases when the optimal trajectory breaks away from the road geometry.

The acceleration term f_{acc} makes the vehicle act smoothly and is defined as

$$f_{acc} = a_{x,i}^2 + a_{y,i}^2 \tag{17}$$

where a_x and a_y are the acceleration in x-axis and y-axis, respectively. This term helps the curvature and the longitudinal acceleration of the final trajectory decrease compared with the reference trajectory of the behavioral one.

The jerk term f_{jerk} is defined as (18).

$$f_{jerk} = j_{x,i}^2 + j_{y,i}^2 \tag{18}$$

smoothenss the trajectory by softening sudden changes in acceleration where j_x and j_y are the jerk in x-axis and y-axis, respectively.

The optimal trajectory consists of discrete parameters, so both acceleration and jerk of the vehicle are also calculated by the difference of parameters as following

$$a_{x,i} = \frac{x_{i+1} - 2x_i + x_{i-1}}{\Delta t^2}, \tag{19}$$

$$a_{y,i} = \frac{y_{i+1} - 2y_i + y_{i-1}}{\Delta t^2}, \tag{20}$$

$$j_{x,i} = \frac{x_{i+2} - 3x_{i+1} + 3x_i - x_{i-1}}{\Delta t^3}, \tag{21}$$

$$j_{y,i} = \frac{y_{i+2} - 3y_{i+1} + 3y_i - y_{i-1}}{\Delta t^3}. \tag{22}$$

2) Constraints: Motion trajectory is in charge of not only minimizing the objective function (15) but also satisfying safety constraints which consist of vehicle and environment constraints. The vehicle constraints mean that the *motion trajectory* should be generated within a vehicle's physical limits for driving safety. To consider them, an acceleration limit is applied based on the circle of traction [37] that keeps the summation of longitudinal and lateral acceleration within the maximum vehicle acceleration:

$$a_{x,i}^2 + a_{y,i}^2 \leq a_{max}^2. \tag{23}$$

These constraints prevent generating a trajectory that the vehicle cannot follow and induce a smooth trajectory by restricting large acceleration. The number of the constraints is equal to N_m which is the number of the motion trajectory's nodes.

Environment constraints refer to the constraints to avoid a collision with dynamic objects. To check the collision with ego vehicle and objects, their boundaries should be computed. However, the boundary of the object is modeled by a discrete function of the bounding box as described in the *planning space*. For an efficient computation of the numerical optimization, the function is converted into the continuous convex function by a circle-based constraint function, as shown in Fig. 9. Since each distance between the ego circles and obstacle's circles is larger than the sum of each circle, the environment constraints are defined as (24)–(27).

$$d_{i,ff} \geq r_{veh} + r_{obs}, \tag{24}$$

$$d_{i,fr} \geq r_{veh} + r_{obs}, \tag{25}$$

$$d_{i,rf} \geq r_{veh} + r_{obs}, \tag{26}$$

$$d_{i,rr} \geq r_{veh} + r_{obs}. \tag{27}$$

The number of environment constraints is four each step time i. Since the number of the motion trajectory's time steps is N_m, the total number of environment constraints can be represented as $4N_{obj}N_m$, where N_{obj} is the number of dynamic objects. Finally, the overall constraints consist of $\left(4N_{obj}+1\right)N_m$ inequations. The inequations can be rearranged by the standard forms $\boldsymbol{g}(\boldsymbol{X}) \geq \mathbf{0}$ of constraint functions for the optimization as

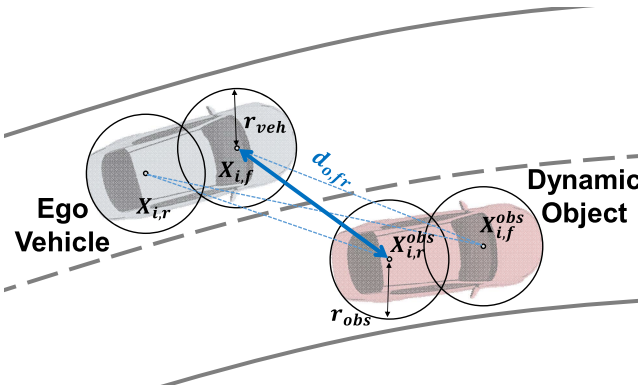

Fig. 9. Environmental constraints to keep a safe distance from dynamic objects.

$$g_k^a(\boldsymbol{X}) = a_{max}^2 - a_{x,i}^2 - a_{y,i}^2, 0 \leq k < N_m \tag{28}$$

$$g_k^d(\boldsymbol{X}) = d_i - r_{veh} + r_{obs}, 0 \leq k < 4N_{obj}N_m \tag{29}$$

$$\mathbf{g}(\boldsymbol{X})^T = \{g_i(\boldsymbol{X})\} = \left[\boldsymbol{g}^{\mathbf{a}}(\boldsymbol{X})\,\mathbf{g}^k(\boldsymbol{X})\right], 0 \leq i < M \tag{30}$$

where g_k^a and g_k^d are the constraints of the acceleration and the collision. M refers to the number of the constraints [$M = \left(4N_{obj}+1\right)N_m$].

3) Boundary Condition: Motion trajectory should keep the maneuver determined by *behavioral trajectory planning*. To guarantee it, behavioral trajectory planning provides the reference as well as the circle-based boundaries of its nodes [red circles in Fig. 8(b)]. Each node of the *motion trajectory* should be generated within the behavioral trajectory boundary. For example, the node T_m^t of the *motion trajectory* should exist in the behavioral boundary B_m^t in Fig. 8(b). Since the trajectory nodes are generated at the interval of the sampling time, the boundary means spatiotemporal boundary.

4) Initial State: An initial state is a key factor to solve the nonlinear optimization problem efficiently. To select an appropriate initial state for the optimization of the *motion trajectory*, the resampled *behavioral trajectory* is used. Since the *behavioral traje*ctory is a local optimal solution in the driving situation, the resampled *behavioral trajectory* is suitable for the initial state to reduce the computation time of the optimization.

5) Trajectory Optimization: A problem to find the motion trajectory is a nonlinear constrained optimization problem that minimizes the objective function (15) and satisfies the constraints and boundary conditions as follows:

$$\underset{\boldsymbol{X}_2, \boldsymbol{X}_3, \ldots \boldsymbol{X}_{N_m-1}}{\operatorname{argmin}} \quad f_{motion}(\boldsymbol{X}_0, \boldsymbol{X}_1, \cdots, \boldsymbol{X}_{N_m-1}), \tag{31}$$

subject to

$$g_i(\boldsymbol{X}) \geq 0, \quad 0 \leq i < M \tag{32}$$

where $g_i(\boldsymbol{X})$ is the constraint functions. The parameters $\boldsymbol{X}_0, \boldsymbol{X}_1$ are the fixed variables that bring the first two values of the previous trajectory for consistency with the previous trajectory.

In this paper, the objective function and the constraint functions are comprised of a quadratic form that is twice

differentiable. To solve the problem, a sequential quadratic programming (SQP) method is applied. This is a method to add the constraint functions into the objective function using Lagrange multipliers $\boldsymbol{L}^{\mathbf{T}} = \left[l_0 l_1 \cdots l_{M-1}\right]$ and to optimize the objective function J through second-order differentiation. The final objective function can be written as,

$$J\left(\boldsymbol{X}, \boldsymbol{L}\right) = f_{motion}\left(\boldsymbol{X}\right) + \boldsymbol{L}^{T}\boldsymbol{g}(\boldsymbol{X}). \tag{33}$$

Basic SQP performs optimization by approximating the objective function using second-order differentiation. The second-order differential matrix, called the Hessian matrix, is updated every iteration using the Broyden-Fletcher-Goldfarb -Shanno (BFGS) update. However, the BFGS update has $O(N^2)$ computational complexity every iteration, requiring a significant amount of computational effort. By calculating true Hessian matrix, the computational effort to find the approximate Hessian matrix is removed.

To calculate this matrix, the first-order derivatives of the objective function can be calculated by

$$\frac{\partial J}{\partial q_i} = \frac{\partial f_{motion}}{\partial q_i} + \sum_{k=0}^{M-1} l_k \frac{\partial g_k(X)}{\partial q_i} \tag{34}$$

where q_i is an element of $\{q_i\} = \{x_1, y_1, x_2, y_2, \ldots, x_{N_m}, y_{N_m}\}$.

The Hessian matrix should be positive definite to find an optimal solution. The objective function and constraint functions are defined as a quadratic form to make the Hessian matrix positive definite. Furthermore, a true Hessian matrix is a sparse and banded matrix because the term of the objective and constraint functions are affected by four or less adjacent parameters. For example, the jerk term f_{jerk} is the sum of the quadratic functions including four adjacent parameters: $\boldsymbol{X_{i-1}}$, $\boldsymbol{X_i}$, $\boldsymbol{X_{i+1}}$, and$\boldsymbol{X_{i+2}}$ as described in (21) and (22) . Since Hessian $\boldsymbol{H}$ is calculated by second-order partial derivatives of the objective function J as follows

$$\boldsymbol{H}_{ij} = \frac{\partial^2 J}{\partial q_i \partial q_j} = \frac{\partial^2 (f_{ref} + f_{acc} + f_{ref})}{\partial q_i \partial q_j} + \frac{\partial^2 \boldsymbol{L}^T \boldsymbol{g}(\boldsymbol{X})}{\partial q_i \partial q_j} \tag{35}$$

The second-order derivative term of the jerk cost is

$$\frac{\partial^2 f_{jerk}}{\partial q_i \partial q_j} = \begin{cases} c_{ij} & ati = j \pm 2k \\ 0 & ati \neq j \pm 2k \end{cases} \tag{36}$$

where the index k of the adjacent parameters is $\{-3, -2, -1, 0, 1, 2, 3\}$. Thus, the matrix of the jerk's second derivative term has only seven lines of diagonal terms, one for every two lines. By using the same process, the Hessian $\boldsymbol{H}$ can be calculated. Since the maximum number of the adjacent parameters is four, the Hessian $\boldsymbol{H}$ also has seven lines of diagonal terms as the jerk's one. It reduces computation effort to compute the Hessian because the diagonal terms are only considered instead of N_m^2.

VI. Experiments

A. Experimental Environments

We evaluated the proposed trajectory planning algorithm by using autonomous car *A1* of Hanyang University in Fig. 10.

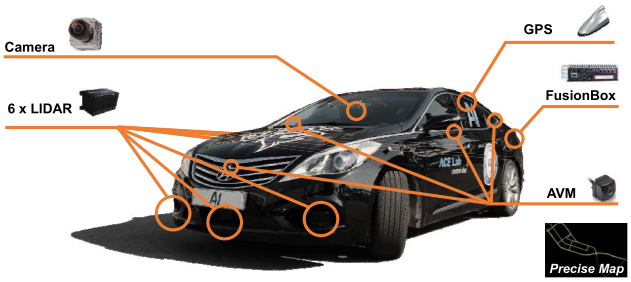

Fig. 10. Test vehicle A1 was equipped with cameras, LIDARs and fusion system, GPS, a precise map to evaluate the trajectory planning.

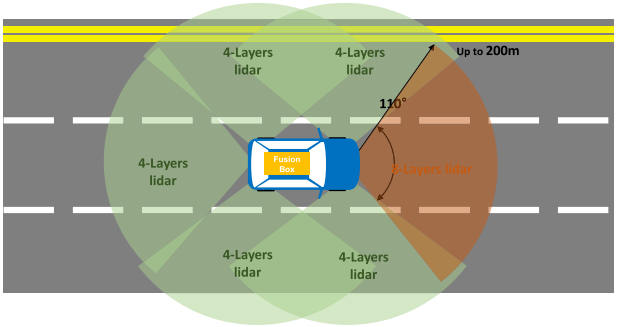

Fig. 11. Region of interest (ROI) to detect the surrounding objects.

This car was equipped with LIDARs and a LIDAR fusion box system, cameras, GPS, and precise digital map.

To detect surrounding objects, six LIDARs and a fusion box system (produced by IBEO) were installed. The region of interest (ROI) of the sensors was configured as shown in Fig. 11. The scan data of multiple LIDARs were synchronized and fused by the fusion box system which provided the point cloud and moving objects. Based on the tracking information of the objects, we generated the *dynamic object* of the *planning space*. In addition, the point cloud was used to extract the occupancy grid map of static objects in the *planning space*.

The test vehicle estimated its precise positon within 0.3 meters using map matching based precise localization system [38]. This system integrated GPS, cameras, LIDARs, and precise map. Through the system, we can obtain the robust estimation of the ego's position in an urban area. In addition, the precise digital map provided the road geometry, lane boundary, traffic signs, stop lines and traffic light information. Based on the information, we generated the *local route* and the *TCD* of the *planning space*. To concentrate on trajectory planning, a global route was predefined.

The proposed planning algorithm was implemented in i5 Core embedded PC in C++ language. The execution periods of the *behavioral trajectory planning* and the *motion trajectory planning* were 50 milliseconds, respectively. All the algorithms were operated in multiple thread environments. Considering the maximum object detection range, the longitudinal range of the searching space was 150 m. The interval Δs is proportional to the vehicle speed and larger than the minimum interval Δs_{min} of $5m$. The step time ΔT for the speed profile in the behavioral trajectory was set to $500ms$. Additional parameters for the motion trajectory planning were

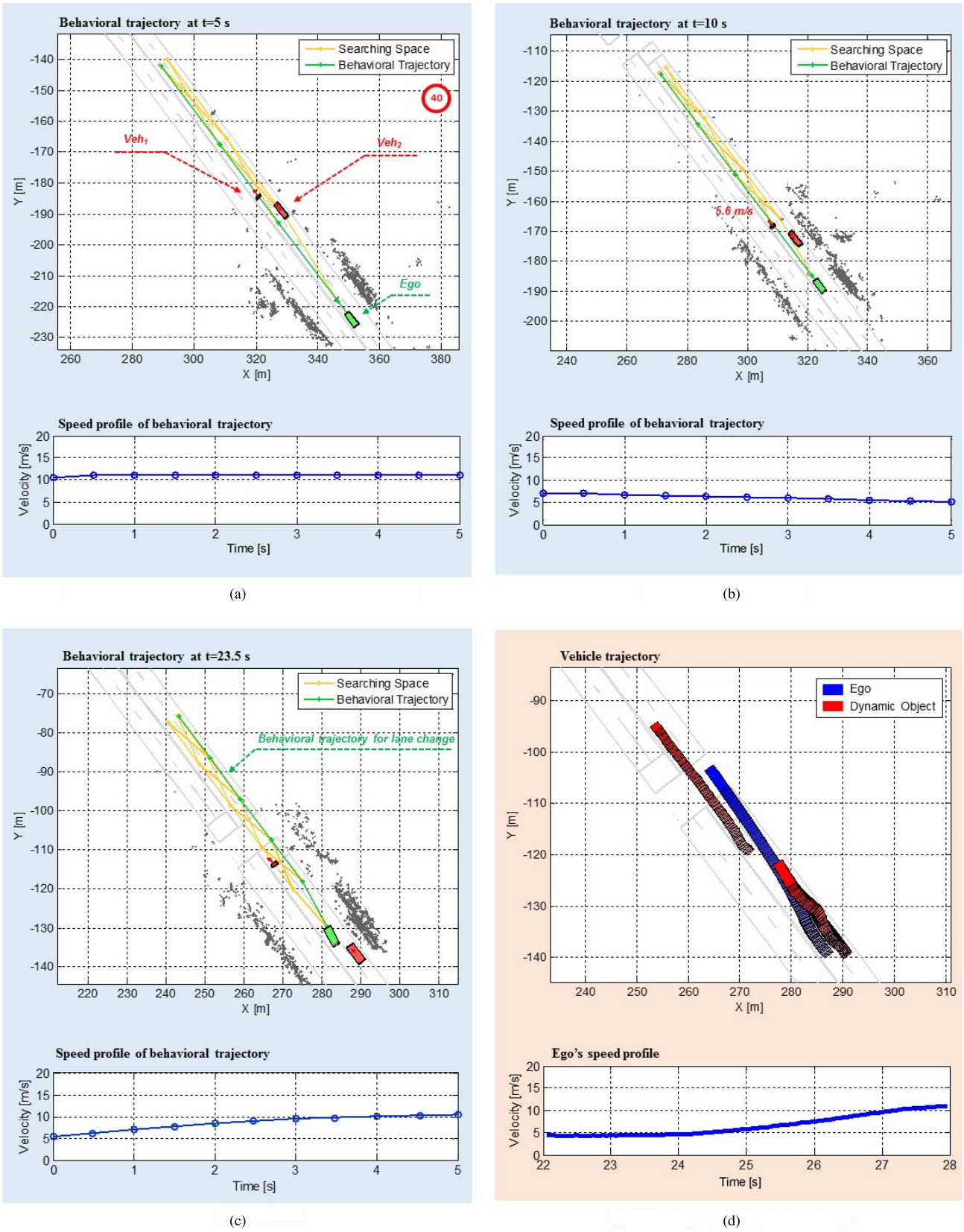

Fig. 12. Evaluation of the behavioral trajectory planning in the lane change scenario. (a) Keeping the speed as the regular speed (40 km/h or 11.1 m/s). (b) Following the front vehicle. (c) Lane change. (d) Vehicle trajectory during the lane change.

set to $\Delta T_{resample} = 100\ ms$ and $N_m = 30$. Furthermore, the vehicle had vehicle control units including the longitudinal and lateral control. The lateral control was based on the modified pure pursuit method with considering the tire slip factors [28]. By using these controllers, the planning algorithm was evaluated in a real driving environment.

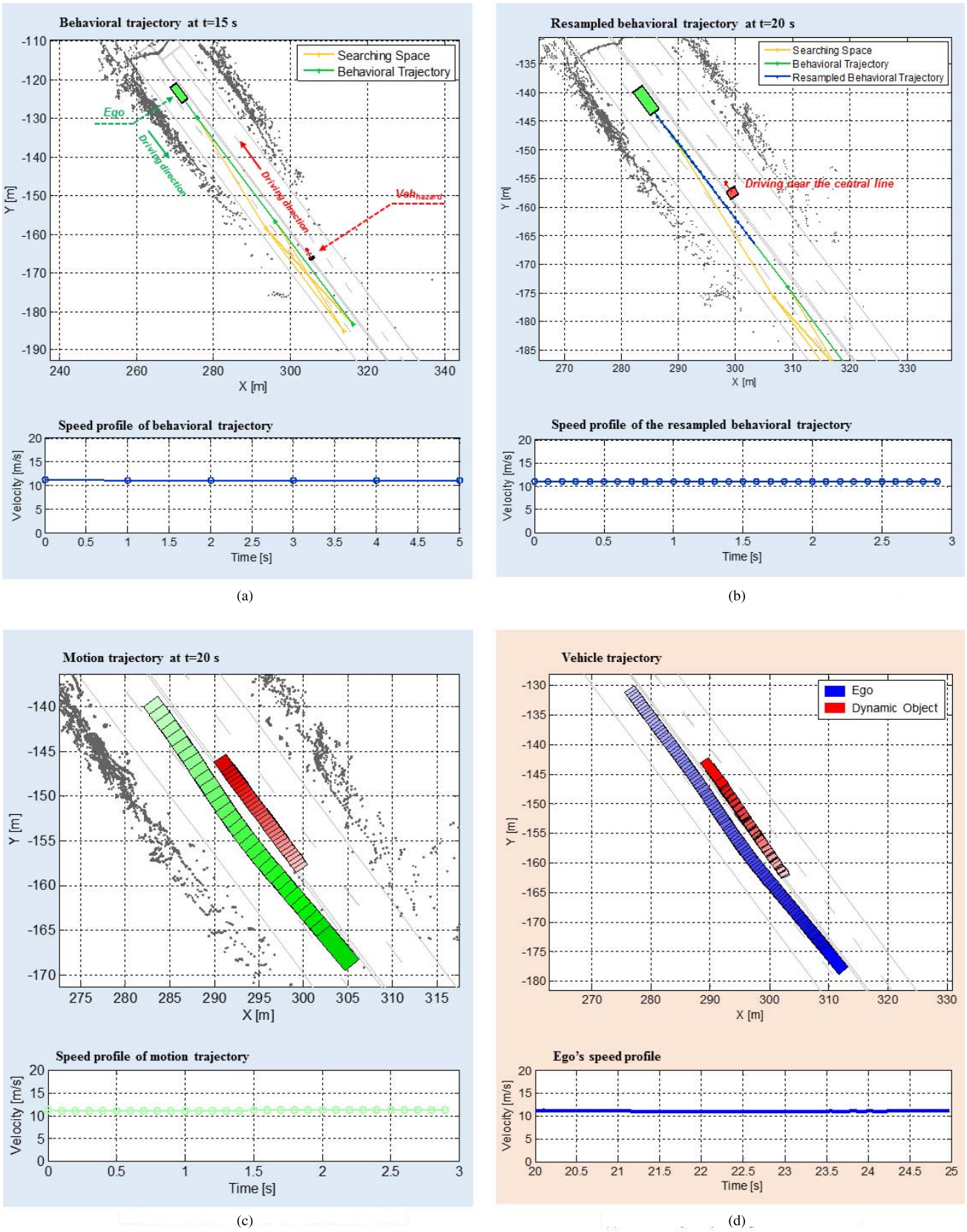

Fig. 13. Evaluation of the motion trajectory planning in the swerve scenario. (a) Approaching of the hazard vehicle in the opposite lane. (b) Resampled behavioral trajectory near the hazard vehicle. (c) Swerve motion trajectory near the hazard vehicle. (d) Vehicle trajectory during the swerve motion.

B. Evaluation of the Behavioral Trajectory Planning

We evaluated the performance of the *behavioral trajectory planning* by using the lane change scenario. In the scenario as shown in Fig. 12, the road has two lanes, and the regulation speed is 40 km/h (11.1 m/s). There were two surrounding vehicles. The first vehicle Veh_1 was a front vehicle which kept

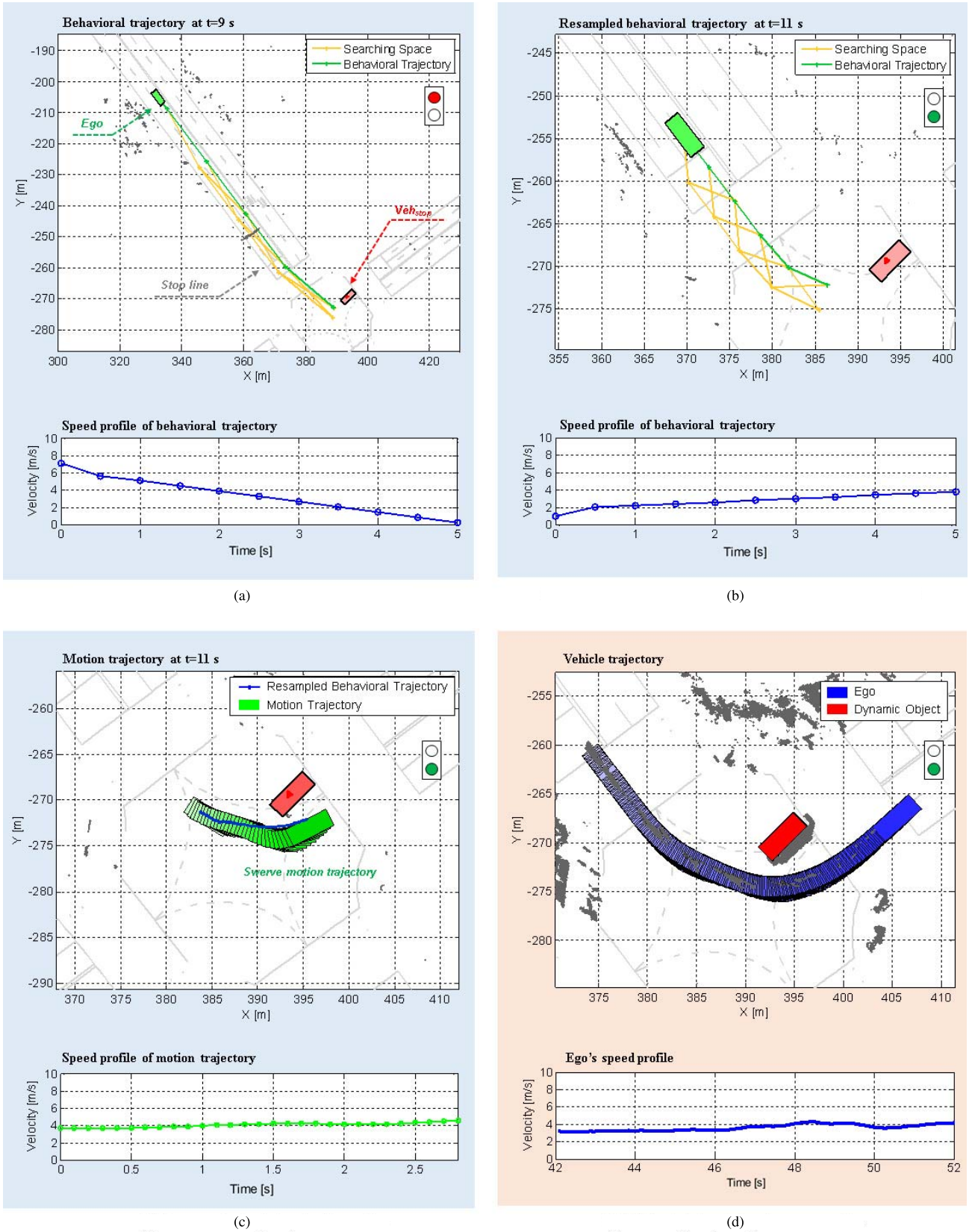

Fig. 14. Evaluation of the integrated trajectory planning in the intersection scenario. (a) Behavioral trajectory for stop when the traffic light is red. (b) Behavioral trajectory for start when the traffic light turns on green. (c) Swerve motion trajectory in a intersection. (d) Vehicle trajectory during the swerve motion.

a constant speed 20 km/h. The other vehicle Veh_2 was driving in the second lane. The initial speed of the vehicle Veh_2 was 20 km/h (5.5 m/s). However, the vehicle started to decrease its speed when the ego vehicle approached the back of the front vehicle. The ego vehicle was placed on the first lane at the first time.

In this experiment, the step time and the maximum distance of searching were 3 seconds and 150 meters, respectively. At t=5 seconds in Fig. 12(a), the *behavioral trajectory* for lane keeping was selected because its lane change cost C_{LC} and travel distance cost C_{dis} were the lowest among the trajectory candidates. The vehicle kept the speed of the regulation speed 40 km/h which is higher than one of the vehicle Veh_1. Thus, the distance between the ego and the vehicle Veh_1 decreased. Since the ego was surrounded by two vehicle Veh_1 and Veh_2, the planner selected the lane keeping trajectory, as shown in Fig. 12(b). However, lane change *behavioral trajectory* was generated when the speed of the vehicle Veh_2 slowed down and the gap between the surrounding vehicles opened up enough to do lane change, as shown in Fig. 12(c). For a lane change, the speed was increased at that time. Finally, the vehicle was driving in the second lane because there was no object in all the lanes. These results were plausible from a driver's point of view, as shown in Fig. 12(d). The *behavioral trajectory planning* provided a sketchy trajectory for the *motion trajectory*. Then, the results of the *behavioral trajectory* bounded the *motion trajectory* problem to find the one local minima. Thus, it helps the functional optimization of the *motion trajectory* to be more easily converged at the local minima.

C. Evaluation of the Motion Trajectory Planning

Motion trajectory was the concrete trajectory more than the *behavioral trajectory* considering the objects and the *TCD* of the *planning space*. It strongly considered the results of the *behavioral trajectory*, vehicle dynamics, and the boundary of objects. We evaluated the algorithm on the swerve scenario, as shown in Fig. 13. There was a hazard vehicle Vhe_{hazard} on the opposite lane. The path adhered to the central line of the road. Although it did not make a collision with ego vehicle, it threatened the passengers in the ego vehicle.

Since the vehicle Vhe_{hazard} did not cross the central line, the *behavioral trajectory* was generated as if there were no objects, as shown in Fig. 13(a)-(b). However, the *motion trajectory* considered the objects due to the safe boundary in (24)–(27). To keep the safe distance to objects, the *motion trajectory* generated the swerve trajectory as shown in Fig. 13(c). The trajectory was optimized based on the numerical optimization method of SQP. The whole trajectory of the ego vehicle was represented in Fig. 13(d).

D. Evaluation of the Integrated Motion Planning

We evaluated the hierarchical trajectory planning in an intersection scenario as shown in Fig. 14. In the scenario, there was a traffic light and a stopped vehicle Veh_{stop} waiting for the green traffic light in the intersection. However, its location was very close to the following line in the intersection.

The performance of the traffic light management was evaluated in Fig. 14(a)-(b). *Behavioral trajectory* generated the stop behavior in front of the stop line because the traffic light of *TCD* was red. Based on the upper trajectory, *motion trajectory* made the vehicle stop as shown in Fig. 14(a). After on the light changed to green, *behavioral trajectory* provided the start maneuver for the *motion trajectory,* as shown in Fig. 14(b). Thus, *behavioral trajectory planning* managed the *TCD* information, and the proper maneuver was selected.

Due to the vehicle Veh_{stop}, the irregular motion trajectory was generated in Fig. 14(c). Although the *behavioral trajectory* made the vehicle follow the predefined road geometry in the intersection, *motion trajectory planning* generated the swerve trajectory for safe driving. The entire trajectory of the ego is plotted in Fig. 14(d).

VII. Conclusion

This paper presents a *planning space* and a hierarchical trajectory planning algorithm for autonomous driving. By integrating the sampling-based and optimized-based trajectory planning, the complexity of driving environments for urban driving and the computation load can be reduced. For the integration, the proposed algorithm has behavioral (upper) and *motion trajectory* (lower). In the *behavioral trajectory planning*, the macro trajectory is generated based on the sampling method considering the discrete information of objects and *TCD*. As a result, the upper planner determines the safe and time-saving maneuver by using sample generation and evaluation. Based on the rough trajectory, the lower trajectory planning algorithm optimizes the local trajectory for safe, energy, and driving comfort by using the numerical optimization method SQP. The performance was evaluated by a real vehicle test. For the evaluation on the road, we provided three scenarios: lane change, swerve, and intersection driving. In each scenario, the proposed planner determined the reasonable maneuver and trajectory. Furthermore, the execution times of these algorithms are within 50 milliseconds, respectively.

In future work, we plan to apply a machine learning algorithm into the behavioral selection for complicated driving scenarios. Pattern adaptation of a searching space into the driving situation could enhance the performance of the proposed algorithm.

References

[1] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A review of motion planning techniques for automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 4, pp. 1135–1145, Apr. 2016.

[2] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Trans. Intell. Veh.*, vol. 1, no. 1, pp. 33–55, Mar. 2016.

[3] M. McNaughton, C. Urmson, J. M. Dolan, and J.-W. Lee, "Motion planning for autonomous driving with a conformal spatiotemporal lattice," in *Proc. IEEE Int. Conf. Robot. Autom.*, Shanghai, China, May 2011, pp. 4889–4895.

[4] S. Wang, "State lattice-based motion planning for autonomous on-road driving," Ph.D. dissertation, Dept. Math. Comput. Sci., Free Univ. Berlin, Berlin, Germany, 2015.

[5] J. Ziegler and C. Stiller, "Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, St. Louis, MO, USA, 2009, pp. 1879–1884.

[6] K. Chu, M. Lee, and M. Sunwoo, "Local path planning for off-road autonomous driving with avoidance of static obstacles," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 4, pp. 1599–1616, Dec. 2012.

[7] T. Gu, J. M. Dolan, and J.-W. Lee, "On-road trajectory planning for general autonomous driving with enhanced tunability," in *Intelligent Autonomous Systems 13*, 1st ed. Cham, Switzerland: Springer, 2016, pp. 247–261.

[8] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for Bertha—A local, continuous method," in *Proc. IEEE Intell. Vehicles Symp.*, Dearborn, MI, USA, Jun. 2014, pp. 450–457.

[9] J. Dickmann, N. Appenrodt, and C. Brenk, "Making Bertha," *IEEE Spectr.*, vol. 51, no. 8, pp. 44–49, Jul. 2014.

[10] C. Urmson *et al.*, "Autonomous driving in urban environments: Boss and the urban challenge," *J. Field Robot.*, vol. 25, no. 8, pp. 425–466, 2008.
[11] J. Kim, K. Jo, K. Chu, and M. Sunwoo, "Road-model-based and graph-structure-based hierarchical path-planning approach for autonomous vehicles," *Proc. Inst. Mech. Eng. D, J. Automobile Eng.*, vol. 228, no. 8, pp. 909–928, 2014.
[12] J. Levinson *et al.*, "Towards fully autonomous driving: Systems and algorithms," in *Proc. IEEE Intell. Vehicles Symp.*, Baden-Baden, Germany, Jun. 2011, pp. 163–168.
[13] M. Brännström, F. Sandblom, and L. Hammarstrand, "A probabilistic framework for decision-making in collision avoidance systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 2, pp. 637–648, Jun. 2013.
[14] J. Ibañez-Guzmán, C. Laugier, J.-D. Yoder, and S. Thrun, "Autonomous driving: Context and state-of-the-art," in *Handbook of Intelligent Vehicles*, 1st ed. London, U.K.: Springer, 2012, pp. 1271–1310.
[15] J. Wei, J. M. Snider, J. Kim, J. M. Dolan, R. Rajkumar, and B. Litkouhi, "Towards a viable autonomous driving research platform," in *Proc. IEEE Intell. Vehicles Symp.*, Gold Coast, QLD, Australia, Jun. 2013, pp. 763–770.
[16] M. Montemerlo *et al.*, "Junior: The Stanford entry in the urban challenge," *J. Field Robot.*, vol. 25, no. 9, pp. 569–597, 2008.
[17] D. Pagac, E. M. Nebot, and H. Durrant-Whyte, "An evidential approach to map-building for autonomous vehicles," *IEEE Trans. Robot. Autom.*, vol. 14, no. 4, pp. 623–629, Aug. 1998.
[18] T. Yang and V. Aitken, "Evidential mapping for mobile robots with range sensors," *IEEE Trans. Instrum. Meas.*, vol. 55, no. 4, pp. 1422–1429, Aug. 2006.
[19] F. Kunz *et al.*, "Autonomous driving at Ulm University: A modular, robust, and sensor-independent fusion approach," in *Proc. IEEE Intell. Vehicles Symp.*, Seoul, South Korea, Jun./Jul. 2015, pp. 666–673.
[20] D. Ferguson, M. Darms, C. Urmson, and S. Kolski, "Detection, prediction, and avoidance of dynamic obstacles in urban environments," in *Proc. IEEE Intell. Vehicles Symp.*, Eindhoven, The Netherlands, Jun. 2008, pp. 1149–1154.
[21] T. Luettel, M. Himmelsbach, and H.-J. Wuensche, "Autonomous ground vehicles? Concepts and a path to the future," *Proc. IEEE*, vol. 100, Special Centennial Issue, pp. 1831–1839, May 2012.
[22] A. Chen, A. Ramanandan, and J. A. Farrell, "High-precision lane-level road map building for vehicle navigation," in *Proc. IEEE/ION Position Location Navigat. Symp.*, Indian Wells, CA, USA, May 2010, pp. 1035–1042.
[23] D. Khosla, "Accurate estimation of forward path geometry using two-clothoid road model," in *Proc. IEEE Intell. Vehicles Symp.*, Versailles, France, Jun. 2002, pp. 154–159.
[24] A. Schindler, G. Maier, and F. Janda, "Generation of high precision digital maps using circular arc splines," in *Proc. IEEE Intell. Vehicles Symp.*, Alcalá de Henares, Spain, Jun. 2012, pp. 246–251.
[25] A. Wedel, H. Badino, C. Rabe, H. Loose, U. Franke, and D. Cremers, "B-spline modeling of road surfaces with an application to free-space estimation," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 4, pp. 572–583, Dec. 2009.
[26] A. Schindler, G. Maier, and S. Pangerl, "Exploiting arc splines for digital maps," in *Proc. Int. IEEE Conf. Intell. Transp. Syst.*, Washington, DC, USA, Oct. 2011, pp. 1–6.
[27] K. Jo and M. Sunwoo, "Generation of a precise roadway map for autonomous cars," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 3, pp. 925–937, Jun. 2014.
[28] K. Jo, J. Kim, D. Kim, C. Jang, and M. Sunwoo, "Development of autonomous car—Part II: A case study on the implementation of an autonomous driving system based on distributed architecture," *IEEE Trans. Ind. Electron.*, vol. 62, no. 8, pp. 5119–5132, Aug. 2015.
[29] K. Kant and S. W. Zucker, "Toward efficient trajectory planning: The path-velocity decomposition," *Int. J. Robot. Res.*, vol. 5, no. 3, pp. 72–89, Jan. 1986.
[30] T. Fraichard, "Trajectory planning amidst moving obstacles: Path velocity decomposition revisited," *J. Brazilian Comput. Soc.*, vol. 4, no. 3, pp. 1–8, Apr. 1998.
[31] J. Kim, K. Jo, W. Lim, M. Lee, and M. Sunwoo, "Curvilinear-coordinate-based object and situation assessment for highly automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 3, pp. 1559–1575, Jun. 2015.
[32] H. Wang, J. Kearney, and K. Atkinson, "Robust and efficient computation of the closest point on a spline curve," in *Proc. 5th Int. Conf. Curves Surf.*, Paris, France, 2002, pp. 397–406.
[33] S. Thrun *et al.*, "Stanley: The robot that won the DARPA Grand Challenge," *J. Field Robot.*, vol. 23, no. 9, pp. 661–692, 2006.
[34] J. Johnson and K. Hauser, "Optimal longitudinal control planning with moving obstacles," in *Proc. IEEE Intell. Vehicles Symp.*, Gold Coast, QLD, Australia, Jun. 2013, pp. 605–611.
[35] J. Johnson and K. Hauser, "Optimal acceleration-bounded trajectory planning in dynamic environments along a specified path," in *Proc. IEEE Int. Conf. Robot. Autom.*, Saint Paul, MN, USA, May 2012, pp. 2035–2041.
[36] H. Choset *et al.*, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Cambridge, MA, USA: MIT Press, 2005.
[37] R. Rajamani, *Vehicle Dynamics and Control*. New York, NY, USA: Springer, 2011.
[38] K. Jo, Y. Jo, J. K. Suhr, H. G. Jung, and M. Sunwoo, "Precise localization of an autonomous car based on probabilistic noise models of road surface marker features using multiple cameras," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 6, pp. 3377–3392, Dec. 2015.

Wonteak Lim (S'14) received the B.S. degree in mechanical engineering from Hanyang University, Seoul, South Korea, in 2013, where he is currently pursuing the Ph.D. degree with the Automotive Control and Electronics Laboratory. His research interests include trajectory planning, vehicle control, real-time embedded systems for intelligent transportation systems, trajectory planning algorithms, system integration, and software platform design for autonomous vehicles.

Seongjin Lee (S'16) received the B.S. degree in automotive engineering from Hanyang University, Seoul, South Korea, in 2015. Since 2015, he has been with the Automotive Control and Electronics Laboratory, Department of Automotive Engineering, Hanyang University, doing research on the implementation of autonomous cars. His research interests include path planning, trajectory planning, and vehicle motion control for highly automated vehicles.

Myoungho Sunwoo (M'81) received the B.S. degree in electrical engineering from Hanyang University in 1979, the M.S. degree in electrical engineering from The University of Texas at Austin in 1983, and the Ph.D. degree in system engineering from Oakland University in 1990.

He joined General Motors Research (GMR) Laboratories, Warren, MI, USA, in 1985. He has involved in automotive electronics and control for 30 years. During his nine-year tenure at GMR, he was involved in the design and development of various electronic control systems for powertrains and chassis. Since 1993, he has been leading research activities as a Professor with the Department of Automotive Engineering, Hanyang University. His research interests include automotive electronics and controls, such as the modeling and control of internal combustion engines, the design of automotive distributed real-time control systems, intelligent autonomous vehicles, and automotive education programs.

Kichun Jo (S'10–M'14) received the B.S. degree in mechanical engineering in 2008, and the Ph.D. degree in automotive engineering from Hanyang University, Seoul, South Korea, in 2014. Between 2014 and 2015, he was with the Automotive Control and Electronics Laboratory, Department of Automotive Engineering, Hanyang University, doing research on system design and implementation of autonomous cars. Since 2015, he has been with Valeo Driving Assistance Research, Bobigny, France, where he has been involved in the highly automated driving. His research interest include localization and mapping, objects tracking, information fusion, vehicle state estimation, behavior planning, vehicle motion control for highly automated vehicles, and hardware and software platform design of autonomous cars based on distributed real-time embedded systems and in-vehicle networks.