2018 IEEE Intelligent Vehicles Symposium (IV)
Changshu, Suzhou, China, June 26-30, 2018

Navigating Automated Vehicle through Expressway Toll Gate

Xuesong Mao[1], Yuquan Xu[1], Seiichi Mita[1*], Hakusho Chin[2] and Hossein Tehrani[2]

***Abstract*— Path planning for automated driving vehicle in wide, unstructured toll gate is a difficult task while several surrounding vehicles are driving with almost random behavior. To deal with this problem, we propose a decision tree based method to control speed & acceleration of the host vehicle along a predefined path. Our proposed method is able to reach the destination without collision and with minimum changes in acceleration or deceleration. To validate the performance, we simulated the toll gate region in Nagoya Expressway, where driving area is wide with no lane markers. Simulation results show that the host vehicle is able to avoid potential collisions through acceleration or deceleration, which is determined by the proposed decision tree. Speed profiles show that host vehicle is driving in a comfortable range of acceleration/deceleration.**

I. INTRODUCTION

Autonomous vehicle navigation in static environment has been extensively studied, and it is no longer a research problem. However, when host vehicle advances toward dealing with realistic road scenarios, they must consider dynamic environments in which information about surrounding obstacles is changing from time to time. Simple maneuvers in structured roads like changing lanes or merging into traffic flow while avoiding other moving vehicles have been already discussed and have shown a good performance. Nevertheless, when traffic scenario becomes complex, previous path planning methods in dynamic environments for vehicles would quickly reach their limits. In this paper, we consider the path planning in toll gate region of an expressway.

To show the complexity of such a problem, we consider a toll gate region of expressway, located at one of the entrances of Tomei expressway in Nagoya, Japan. It includes six automated and manual entrance toll gates and two ramps for leading vehicles to the main roads. Vehicles getting out of any toll gates are likely to direct to either of the two ramps. Additionally, each ramp has two lanes, which further increases the complexity of feasible vehicle paths. Figure1 shows the map of Tomei expressway toll gate in Nagoya, along with the possible paths connecting toll gates and ramps. The exit of the toll gate is 30m wide and there are no lanes markers as shown in Fig. 1. Each designed path has at least 20 intersections with other paths, showing complexity of the driving scenario in Expressway toll gate. Speed and direction of surrounding vehicles could be varying, causing more complicated scenarios for planning in such a dynamic environment.

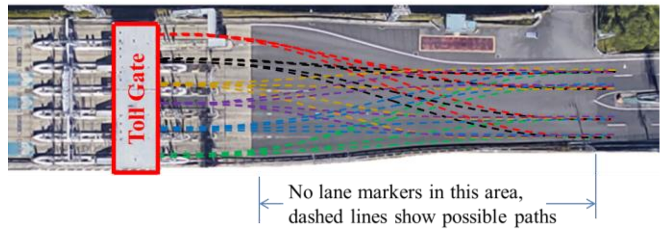


Fig. 1. A toll gate region of expressway in Nagoya, Japan. An example of a wide unstructured area at toll gate. No-lane marking situation allows vehicles to drive in any direction causing complex traffic situation.

For automated vehicle path planning in unstructured environment, many methods have been already proposed in the literature, such as the occupancy grid map based searching algorithms including A* [1], D*[2], and anytime dynamic A* heuristic search [3]. Their Simplicity and ability to find the shortest path must be remarked. However, these methods require to build a grid map before searching and we need to make a big size grid map to cover toll gate environment. The choice of grid map size may be very different when environment changes, which makes the planning problem even more difficult for complex environments such as toll gate. Besides, the resultant path should be smoothened to get a continuous path.

Another approach of path planning is the graph-based method, including state lattice [4-5], visibility graph [6-7], and map knowledge [8-9]. In these methods, nodes along the road surface are created and then they are connected to make a topological graph and the creation of nodes would be heavy when the number of obstacles on the road are large. State lattice considers combined state of heading, steering and position, that increases the number of nodes to a tremendous amount for wide, unstructured spaces such as toll gate area. In fact, the visibility graph is not achievable in dynamic environments when surrounding vehicles are moving randomly. Knowledge based methods split narrow road into regions to create an undirected graph, which also deals with only static environments.

Other typical path planning methods that do not rely on the environment map, include maximum margin based method [10-11] and artificial potential field (APF) [12-13]. Both mentioned methods obtain optimal path basing on road boundaries and obstacle positions in the road. The maximum margin based method categorizes the points that are does not cross obstacle and applies support vector machine to generate a collision free corridors for vehicle movement. APF method defines potential functions for various types of obstacles, and then plans a path by moving in the descent gradient direction of the total potential field. For a wide and unstructured space such as toll gate area, the generated paths which connect all

*Corresponding author. smita@toyota-ti.ac.jp
1. Research Center for Smart Vehicles, Toyota Technological Institute, 2-12-1 Hisakata, Tempaku, Nagoya, Aichi, 468-8511, Japan, {xsmao, yuquan.xu86}@toyota-ti.ac.jp
2. Corporate R&D1, DENSO CORPORATION, 1-1, Showa cho, Kariya, Aichi, 448-8661, Japan, {hakusho_chin, hossein_tehrani}@denso.co.jp

start and end positions, converge to the central line in the middle part of the space and these paths are not optimized either in curvature or length optimal path.

The classical path planning algorithms in the literature, have challenges under wide and unstructured toll gate area either in heavy calculation cost or generating smooth behavior for host vehicle. In order to achieve safe driving, we propose a decision tree based method that requires less calculation cost and generates smooth human-like behavior in Expressway toll gate area. We propose a decision tree to generate smooth behavior by setting near collision points as its nodes. The near collision points are defined as the coordinate where collision between the host vehicle and the surrounding vehicle would happen within certain amount of time. From proposed decision tree, we optimize the action plan of the host vehicle whether to accelerate or decelerate.

In order to achieve low calculation cost and smooth human-like behavior, we propose a method that the host vehicle navigates to the goal as fast as possible, using minimum changes in acceleration and deceleration considering the maneuvers of several surrounding vehicles at same time. Furthermore, maximum allowed acceleration and decelerations are defined to keep the comfort of generated behavior. The simulation results show that the proposed method can safely navigate the host vehicle from start to destination in wide and unstructured toll gate area with dynamic surrounding vehicles, if the velocity and position of surrounding vehicles are correctly estimated in every time step.

The rest of the paper is organized as follows: In Section II, the method to calculate an optimized reference path for the host vehicle is briefly described. In section III, a collision avoidance criteria is defined according to human driving experience. Then, we generate action plan of the host vehicle to accelerate or decelerate, which is determined from the proposed decision tree. In Section IV, we show the simulation result of the proposed method to safely navigate the host vehicle through complex and dynamic Expressway toll gate area. Finally, in section V, the proposed method is concluded.

II. GENERATE PATH

Safe driving in unstructured environment such as toll gate areas is a difficult task since the direction and speed of surrounding vehicles is changing dynamically. In order to address this problem, vehicle navigation is divided into two steps. At first, we generate a predefined path that has a safe range from static obstacles including road boundaries to connect the start and end position, while ignoring the dynamic surrounding vehicles. In the second step, we control the acceleration and deceleration of host vehicle along the path to avoid collisions with dynamic surrounding vehicles. We choose to solve the following six order polynomial equation to generate a path from start to end.

$$y = [a_0 \quad a_1 \quad \cdots \quad a_6] \cdot [1 \quad x \quad \cdots \quad x^6]^T, \tag{1}$$

in the same way as described already in [16]. Coefficients from a_0 to a_6 are calculated under the constraints of host vehicle initial states

$$\left(x, y, \left.\frac{dy}{dx}\right|_{x=x_0}, \left.\frac{d^2 y}{dx^2}\right|_{x=x_0} \right), \tag{2.1}$$

and final states

$$\left(x, y, \left.\frac{dy}{dx}\right|_{x=x_f}, \left.\frac{d^2 y}{dx^2}\right|_{x=x_f} \right), \tag{2.2}$$

where (x, y) stands for the center position of vehicles rear axis in global coordinate system, dy/dx and d^2y/dx^2 are related to the heading angle and the steering angle, respectively. The values of these constraints can be obtained by reading the digital information of the toll gate map. Seven degrees of freedom exist in equation (1) and six constraint equations can be obtained by substituting equation (2) into equation (1). We choose path curvature as the optimization criterion, since smoothness and safety are more concerned than other factors, such as length, or energy saving, among others.

III. DECISION TREE FOR COLLISION AVOIDANCE

We assume that all vehicles accelerate from toll gates to the ramps and each vehicle has an initial speed from 0km/h to 20km/h, depending on manual or automated toll gate when a vehicle exits from the gate. We also assume that the road surface is dry and the weather condition is ideal. Under these conditions, we consider a minimum 3s time to recognize a collision hazard and respond safely [17]. Here, we define t_c as maximum allowed time to collision with surrounding vehicles. In the proposed method, we use human driving experience to make a decision tree to navigate the host vehicle through the predefined optimal path with dynamic objects safely. We consider random behavior of surrounding vehicles to accelerate and decelerate to check the effectiveness and robustness of the method.

A. Problem Definition

Host vehicle drives along the predefined path that is generated in Section II. If only one surrounding vehicle presents in the driving scenario, the host vehicle can simply control its speed to avoid collision by predicting the near collision point along its path. As is shown in Fig. 2, host vehicle estimates the position of *near collision point* based on the speed and heading direction of surrounding vehicle, and then calculates the time t_{12}, which is the required time by the surrounding vehicle to arrive the near collision point.

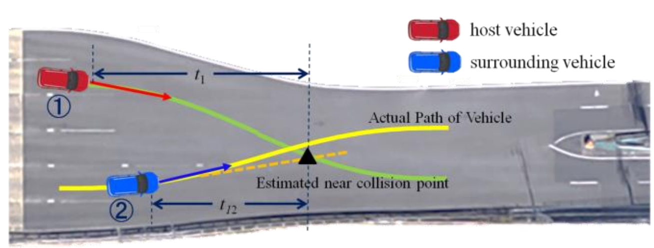


Fig. 2. A single surrounding vehicle situation, a black triangle shows the near collision point between the host vehicle and the surrounding vehicle, which is estimated from the current velocity vector of the vehicle.

Notice that the estimated near collision point at each time instant might be different from the actual collision point. We draw orange line in the Fig. 2 to show the estimated path of vehicle ② based on the speed and heading direction, which in fact is a little different from the real path of vehicle ② showed by yellow curve. If t_{12} is less than t_c (maximum allowed time to collision), the host vehicle decelerates to let the surrounding vehicle ② pass the near collision point first. If t_{12} is longer than t_c, the host vehicle judges whether it can pass the near collision point by acceleration or not. If the host vehicle can pass the near collision point, it accelerates and passes it, otherwise, it decelerates to let the surrounding vehicle to pass first.

When surrounding vehicles are increased to two and more, the problem becomes much more complicated. As shown in Fig. 3, there are two near collision points on the path of the host vehicle. Not only the distance of host vehicle to near collision points, but also the interval of two near collision points should be considered for safe navigation. The host vehicle is dealing with following scenarios:

1) The host vehicle can pass both near collision points by acceleration;

2) The host vehicle can pass the first near collision point but cannot pass the second near collision point;

3) The host vehicle can pass the second near collision point but cannot pass the first near collision point; and

4) The host vehicle cannot pass any near collision points. Compared to the previous situation, the host vehicle should generate more complicated behavior to deal with these complex situations.

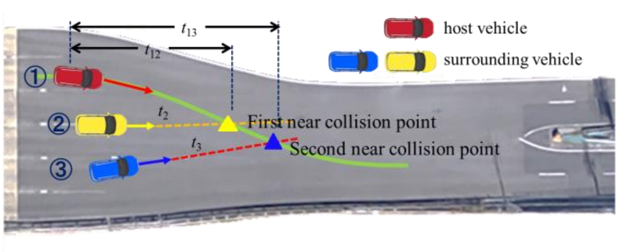


Fig. 3. Two surrounding vehicles situation. t_2 and t_3 stand for the time required by surrounding vehicles to arrive at their respective near collision points. t_{12} and t_{13} stand for the time required by host vehicle to arrive at first and second near collision point, respectively.

When the number of surrounding vehicles are increasing in real practical scenarios, the complexity and difficulty of navigating the host vehicle increases exponentially. In the next section, we propose a method that can navigate the host vehicle under complex situation when there are several surrounding vehicles with dynamic behavior.

B. Decision Tree with Unpredictable Number of dynamic surrounding Vehicles

We consider the situation that *N* number of near collision points occur on the path of host vehicle from toll gate to destination. We assume that positions and speeds of all surrounding vehicles are estimated by vehicle sensors or through V2V communication. All near collision points, pnc_i ; $i = 1, \ldots . N$ are predicted by current position and speed of surrounding vehicles. In addition, the times required by surrounding vehicles to arrive at the near collision points are estimated using current speed of host vehicle. To reduce the computation load, only those points that surrounding vehicles will arrive at in a time less than t_c, (for example, t_c = 5s) will be picked up. At every time step, the newly predicted or disappeared near collision points should be added to or deleted from the processing queue. Next, the elements in the queue are arranged according to their distance to the host vehicle, sorting pnc_i into pnc_k from the shortest to the longest distance from the host vehicle. Two status variables, Acc and Dec, are assigned to each near collision point, which are defined as '*accelerate to pass*' and '*decelerate to give way*', respectively. Each variable has two values, equal to 0 or 1, indicating the whether the host vehicle can avoid collision or not. Zero means collision cannot be avoided at that point by corresponding action, while one means that we can avoid the collision point by selected action.

A path that has multiple near collision points is shown in Fig. 4, along with the assignment of status variables.

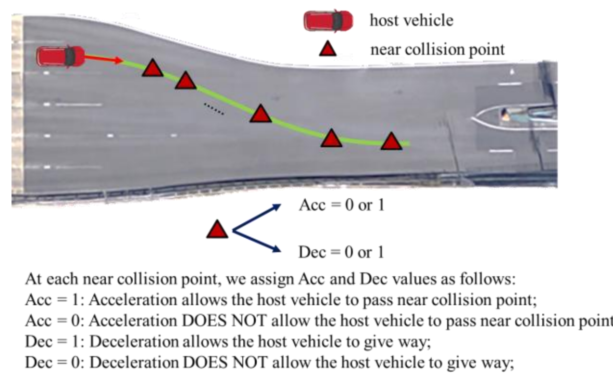
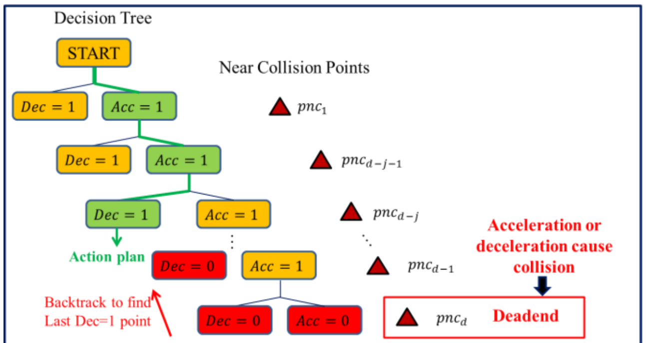


Fig. 4. *N* number of near collision points existing on the path of host vehicle from toll gate to ramp.

If both status variable Acc and Dec of some points in the queue equal to zero, it indicates the corresponding near collision point cannot be avoided either through acceleration or deceleration. On the other hand, if status variable Acc of all points in the queue equal to one, the host vehicle can drive through the path safely by acceleration, without the need to consider the value of the status variable Dec.

In order to determine the action plan of the host vehicle, we consider two possible actions of acceleration and deceleration at each near collision point. We create the decision tree by connecting these points as shown in Fig.5. In order to reach the destination in the minimum time, we only extend $Acc = 1$ branchs. Consider the case shown in Fig.5 as a constructed decision tree. The decision tree stops extending when it reaches the *dead end* shown by pnc_d, where the acceleration or deceleration is no longer possible or we reach to the last collision point. Then, we back-track in the tree j steps to search the last near collision point that has $Dec = 1$, shown as

pnc_{d-j} in Fig. 5. Then the generated action plan of the host vehicle is to accelerate from pnc_1 to pnc_{d-j-1} then to decelerate at pnc_{d-j} . As such, the action plan of the Fig.5 case is defined as "The host vehicle passes first two vehicles by acceleration and give way to third vehicle." In the Fig.6, the pseudo code of the algorithm is shown.



STEP1: Create a sorted decision tree based on time to reach near collision points
STEP2: At each near collision points, determine if the host vehicle can pass the point with acceleration or deceleration
STEP3: Extend tree only to Acceleration until the dead end reached
STEP4: Backtrack to find last $Dec = 1$ point
STEP5: Determine an action plan (shown in green) and calculate required acceleration and deceleration to perform this action plan.

Fig. 5. The decision tree construction and action plan selection

```
for Every Time Step t_s
  for Every Surrounding Vehicle
      Calculate Near Collision Point pnc_i ; i = 1,....N, N = Number of Surrounding Vehicles
      Sort Near Collision Point P_nc,i based on their distance to host vehicle
                              {pnc_k|k = 1,.....N}
  end
  for pnc_1,.....pnc_k,.....pnc_N
    If Time to reach Near Collision Point k > t_c (Maximum Allowed Time to Collision) : Acceleration Case
      if host vehicle can pass the vehicle k with maximum allowed acceleration a_max
          Acc set to be 1 in decision tree
          calculate and store minimum acceleration (m/s^2) required and set as acceleration value
      else
          Acc set to be 0 in decision tree
          set current near collision point pnc_k as dead end pnc_d
          Find last near collision point pnc_d-j that has Dec = 1
          break;
      end
    else : Deceleration case
      if host vehicle can give away the vehicle k with maximum allowed deceleration a_min
          Dec set to be 1 in decision tree
          calculate and store minimum deceleration (m/s^2) required and set as deceleration value
      else
          Dec set to be 1 in decision tree
      end
    end
  end

  : Generate Action Plan from Decision Tree
  Find maximum value from stored acceleration values and  set as host vehicle acceleration a_host
  Set Action plan, the host vehicle accelerates at a_host from pnc_1 to pnc_d-j-1
  Set Action plan, the host vehicle decelerates at deceleration value at pnc_d-j
end
```

Fig. 6. The pseudo code representation of the proposed algorithm

This proposed decision tree based method allows the host vehicle to accelerate and pass several vehicles at once, to minimize the time. It also minimize the changes in acceleration or deceleration to keep the comfort of driving by given the predefined path and the decision tree only extending to Acc to reduce the calculation time of proposed method. Furthermore, we repeat and update this calculation with certain frequency to generate and update the decision tree thus keeping up with the change in velocities or behavior of surrounding vehicles.

IV. Simulation Results

In order to validate the proposed method, we recreated by simulation the toll gate in Nagoya expressway, as shown in Fig. 7. In the simulation, all the vehicles (including the host vehicle) choose random paths from the combination of the nodes associated to four start lanes, six entrance toll gates, and four exit destination lanes, represented by the orange dots in Fig. 7. The total of 96 possible paths accounts for the potential complex traffic situations of surrounding vehicles at the toll gate area. In every simulated iteration, we randomized the initial speed of all vehicles from 0 km/h to 20 km/h, with random acceleration and deceleration between 1 m/s^2 to -2.5 m/s^2, and weset the velocities of all vehicles between 0 km/h and 60 km/h at any time. All the vehicles drive under the restriction of the vehicle kinematic model and only the host vehicle is controlled by the proposed method.

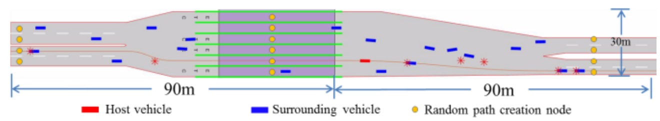


Fig. 7. Recreation of Nagoya expressway toll gate in the simulation.

In this simulation, various driving scenarios are generated from randomized conditions to validate the performance and robustness of the proposed decision tree based method. We ran our MATLAB coded program for more than 100 times, and successfully reached the destination on every occasion, showing the robustness of the method. As a showcase, the following three scenarios are proposed, focusing on the toll gate exit area.

In these three scenarios, the surrounding vehicles path, initial speed and maneuver (acceleration/deceleration) are selected randomly to simulate different driving patterns at toll gate. The proposed method successfully generates different action plans for each scenario, depending on the surrounding vehicle maneuvers.

Figure 8 shows the results of the decision tree for maneuvering the host vehicle in Scenario A. As shown in Fig. 8, the host vehicle selects the *deceleration* to avoid collision as the host vehicle is unable to pass the nearest near collision point by acceleration. In scenario A, the generated action plan is "Decelerate at First Near Collision Point".

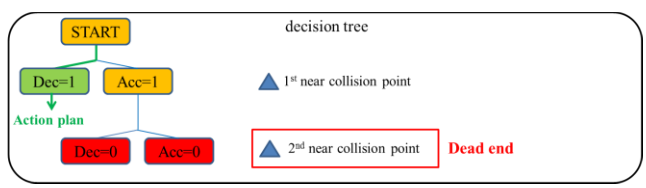
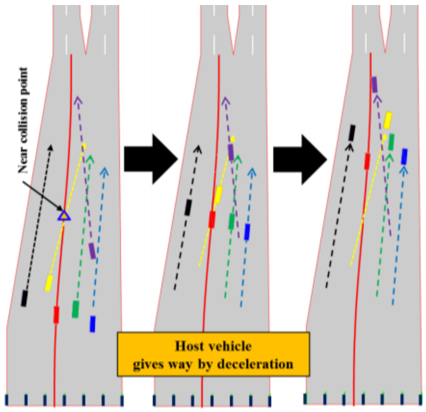
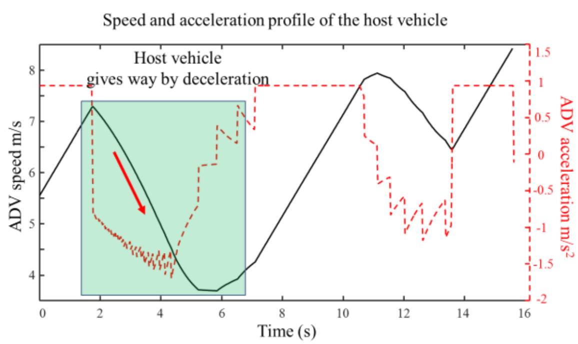


Fig. 8. The decision tree and an action plan of the scenario A.

The path of the host vehicle connecting the start and end position is shown by the red line in Fig. 9(a). At every time step, the host vehicle predicts near collision points on its path according to the speed and heading of the surrounding vehicles. If the surrounding vehicles cross the path within *the maximum allowed time to collision* t_c (in this simulation the t_c = 5 seconds), the predicted near collision point will be added to a queue in an order according to its distance to the host vehicle. In Fig. 9(a), we use rectangles to represent vehicles and arrows to represent their trajectories. The red rectangle represents the host vehicle, and the other colors represent the surrounding vehicles. As shown, the host vehicle successfully predicts the near collision points (shown by the orange triangle) according to the speed and position of the surrounding vehicles at every time step. According to the results of the proposed decision tree, the host vehicle reduces its speed to give way to the yellow vehicle that is merging into its path. The speed and acceleration profiles of the host vehicle are shown in Fig. 9(b).



(a)



(b)

Fig. 9 Scenario A sequence where the host vehicle reduce its speed to give way to the surrounding vehicles. (a) Path of the host vehicle and position of the surrounding vehicles in three time steps; (b) Speed and acceleration profile of host vehicle from the toll gate until the exit ramp.

Figure10 shows the results of the decision tree for maneuvering the host vehicle in Scenario B. As shown in Fig. 10, the host vehicle finds that the first car can be passed by acceleration but decelerates to give way to the second car. In scenario B, the generated action plan is accelerate at first near collision point then to decelerate at second near collision point."

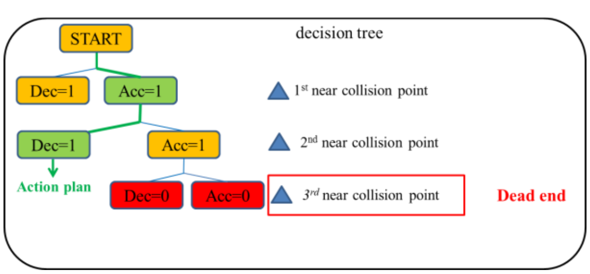
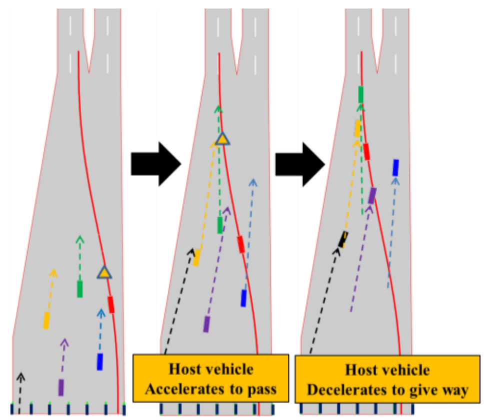
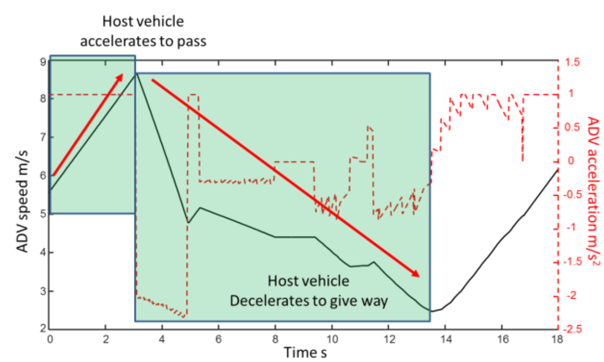


Fig. 10. The decision tree and an action plan of the scenario B

The path of the host vehicle is shown in Fig.11(a). In the first time step, the host vehicle detects a near collision point, which can be passed without collision through acceleration. After passing the first near collision point, the host vehicle decelerates since the second collision point cannot be avoided through acceleration. The host vehicle speed and acceleration profiles are shown in Fig.11(b).



(a)



(b)

Fig. 11. Scenario B where the host vehicle accelerates to pass the first near collision point since it judges that it can pass safely. (a) Path of the host vehicle and two moving steps of the host vehicle and its surrounding vehicles; (b) Speed and acceleration profiles of the host vehicle from the toll gate to the ramp.

Scenario C is the case where the host vehicle finds that all the near collision points can be passed through acceleration from the start to the end. The decision tree is shown in Fig.12. The generated action plan is "Accelerate at first to the second near collision point." Note that the third near collision point is the imaginary point added to complete the algorithm, but the deceleration at the third near collision point will not be performed. The simulation results are shown in Fig.13. The path of the host vehicle is shown in Fig.13(a). The host vehicle detects near collision points along the path, but recognizes that it can pass all of them safely. Figure 13(b) shows the speed and acceleration profiles of the host vehicle. Since all near collision points can be passed, the host vehicle drives toward its destination with the maximum preset acceleration till it reaches a speed of 60km/h, which is the limited speed on the ramp.

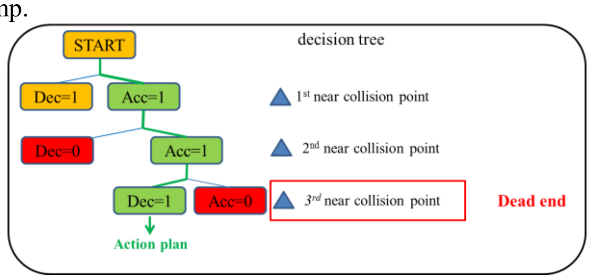
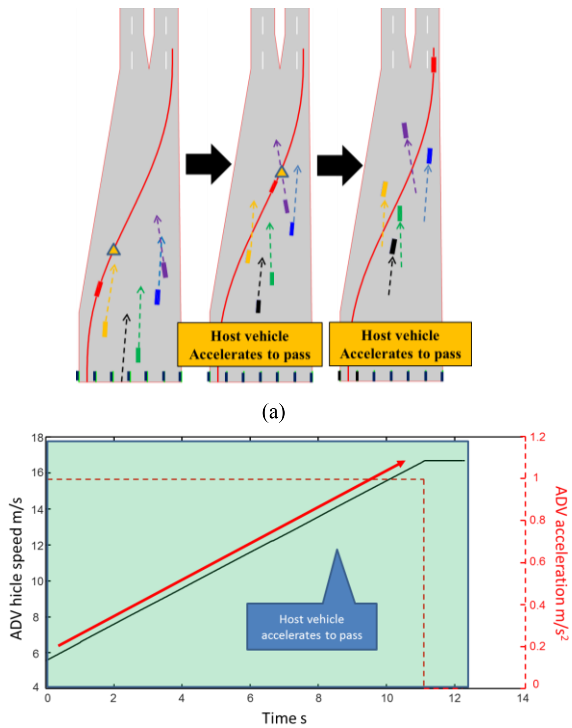


Fig. 12. The decision tree and an action plan of the scenario C.



(a)

(b)

Fig. 13. Scenario C where the host vehicle accelerates to pass all near collision points along the path. (a) Path of the host vehicle and two moving steps of the host vehicle and its surrounding vehicles; (b) Speed and acceleration profiles of the host vehicle from the toll gate to the ramp.

V. CONCLUSION AND FUTURE WORK

In this work, we have proposed a decision tree based method for controlling the speed of the host vehicle in order to safely drive across the wide and unstructured toll gate region with complex traffic situation. Our proposed method has low calculation cost and more human-like behavior compared to other classical path planning methods, due to the predefined path and limited extension of the decision tree. Our decision tree based method allows the host vehicle to accelerate to pass several vehicles at once, resulting in minimum changes in the acceleration or deceleration to maintain the comfort of passengers with safely reaching the destination in the fastest manner. We validated robustness of our method through various simulation scenarios recreating the complex traffic situation of the toll gate. The results showed that the proposed method correctly prevents the host vehicle colliding with surrounding vehicles through deceleration or acceleration, in all simulated scenarios. For the future work, we consider improving the limitation of the current algorithm by implementing path correction algorithm for those cases in which a static object is detected on the predefined path. In order to further consider the application of this algorithm, we also consider applying our method to similar complex scenarios, such as a multilane intersection or a parking area with heavy traffic.

REFERENCES

[1] A. K. Guruji, H. Agarwal, D. K. Parsediya, "Time-efficient A* algorithm for robot path planning," Procedia Technology, vol. 23, 2016, pp. 144-149.

[2] M. Dakulovic, and I. Petrovic, "Two-way D* algorithm for path planning and replanning, Robotics and Autonomous Systems," vol. 59, no. 5, 2011, pp. 329-342.

[3] M. Likhachev, .D. Ferguson, G. Gordon, A. Stentz, and S. Thrun, "Anytime dynamic A*: an anytime, replanning algorithm," in Proceedings of 15th International Conference. on Automated Planning and Scheduling, Monterey, 2005, pp. 262-271.

[4] D. Madas, M. Nosratinia, M. Keshavarz, P. Sundstrom, R. Philippsen, A. Eidehall and K.M. Dahlen, "On path planning methods for automotive collision avoidance," in Proceedings of. Intelligent Vehicles Symposium, IEEE, Gold Coast, QLD, 2013.

[5] M. Cirillo, "From videogames to autonomous trucks: a new algorithm for lattice-based motion planning," Intelligent Vehicles Symposium, IEEE, Redondo Beach, 2017.

[6] M. E. Khaili, "Visibility graph for path planning in the presence of moving obstacles," IRACST-Engineering Science and Technology: An International Journal (ESTIJ), vol.4, no.4, 2014, pp. 118-123.

[7] J. D. Contreras, F. Martinez S, F. H. Martinez S., "Path planning for mobile robots based on visibility graphs and A* algorithm," in Proceedings of. 7th International Conference on Digital Image Processing, SPIE, Los Angeles, 2015, vol. 9631.

[8] Q. H. Do, and S. Mita, "Enhanced path planning in urban environment using knowledge base," in Proceedings of 4th International Symposium on Future Active Safety Technology Toward Zero Traffic Accidents, SAE of Japan, Nara, 2017.

[9] Y. Chen, L. Cheng, H. Wu, X. Zhao, J. Han, "Knowledge-driven path planning for mobile robots: relative state tree," Soft Computing, vol. 19, 2015, pp. 763-773.

[10] Q. H. Do, S. Mita, H. T. N. Nejad, L. Han, "Dynamic and safe path planning based on support vector machine among multi moving obstacles for autonomous vehicles," IEICE Transactions on Information and Systems, vol. E96-D, no. 2, 2013, pp. 314-328.

[11] J. Miura, "Support vector path planning," in Proceedings of International. Conference on Intelligent Robots and Systems, IEEE/RSJ, Beijing, 2006, pp. 3435-3440.

[12] Y. Rasekhipour, A. Khajepour, S. K. Chen, and B. Litkouhi, "A potential field-based model predictive path-planning controller for autonomous road vehicles," IEEE transactions on Intelligent Transportation Systems, vol. 18, no. 5, 2017, pp. 1255-1267.

[13] J. Ji, A. Khajepour, W. W. Melek, .Y. Huang, "Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints," IEEE Transactions on Vehicular Technology, vol. 66, no. 2, 2017, pp. 952-964.

[14] C. Hubmann, M. Becker, D. Althoff, D. Lenz, and C. Stiller, "Decision making for autonomous driving considering interaction and uncertain prediction of surrounding vehicles," in Proceedings of Intelligent Vehicles Symposium, IEEE, Redondo Beach, 2017.

[15] Y. Yang, L. Zhang, X. Qu, J. Lei, Y. Li, and J. Wang, "Smooth path planning for autonomous parking system, in Proceedings of Intelligent Vehicles Symposium," IEEE, Redondo Beach, 2017.

[16] J. Yang, Z. Qu, J. Wang, and K. Conrad, "Comparison of optimal solutions to real-time path planning for a mobile vehicle," IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans, vol. 40, no. 4, 2010, pp. 721-731.

[17] https://www.travelers.com/resources/auto/travel/3-second-rule-for-safe-following-distance.aspx.