Intelligent Intersection Management of Autonomous Traffic Using Discrete-Time Occupancies Trajectory

Qiang Lu and Kyoung-Dae Kim
Department of Electrical and Computer Engineering, University of Denver, Denver, USA
Email: {Qiang.Lu, Kyoung-Dae.Kim}@du.edu

Abstract—**In this paper, a novel approach, called the *Discrete-Time Occupancies Trajectory (DTOT),* is proposed for intersection management of autonomous traffic in which all vehicles in the system are Autonomous Vehicles (AVs) and capable of wireless Vehicle-to-Intersection (V2I) communication. The main advantage of the proposed DTOT-based intersection management is that it enables us to utilize the space within an intersection more efficiently resulting in less delay for vehicles to cross the intersection. In the proposed framework, an intersection coordinates the motions of AVs based on their proposed DTOTs to let them cross the intersection efficiently while avoiding collisions. In case when there is a collision between vehicles' DTOTs, the intersection modifies conflicting DTOTs to avoid the collision and requests AVs to approach and cross the intersection according to the modified DTOTs. We verify the efficiency of the proposed approach through simulation using an open-source traffic simulator, called the Simulation of Urban Mobility (SUMO). To correctly capture the throughput and fairness of algorithms, we also defined a novel measure: *Effective Average Trip Time*. The simulation results show that DTOT algorithm performs better than other existing intersection management scheme: concurrent intersection management scheme.**

Index Terms—**Discrete - Time occupancies trajectory (DTOT), autonomous vehicles, intelligent intersection management, Intelligent Transportation System (ITS)**

I. INTRODUCTION

Advances in sensing and computation technologies have further spurred interest in autonomous driving and many efforts have been made in the past decade. Among many research problems toward the realization of autonomous traffic system, improving throughput while ensuring safety of the traffic is the most important issue that needs to be considered. To address such an issue, a number of notable frameworks have been proposed in the literature [1]-[11].

In reference [1], a scheme that consists of a time-slot allocation intersection crossing algorithm, and an algorithm for updating failsafe maneuvers of each vehicle so as to avoid collision while crossing an intersection was proposed. Reference [2] proposed Cooperative Vehicle Intersection Control (CVIC) algorithm which can manipulate individual vehicles' maneuvers by providing them proper acceleration or deceleration rate in order that vehicles can safely cross the intersection. Jia Wu *et al.* [3] proposed an ant colony system algorithm which is able to solve the combinatorial optimization problems of traffic control in real time. Simulation results show that the algorithm outperforms traditional traffic lights and other recent traffic control strategies. Reza Azimi *et al.* proposed reliable intersection protocols managing traffic through junctions and roundabouts only using Vehicle-to-Vehicle (V2V) communications [4]. In their paper, the intersection area is considered as a grid which is divided into small cells. Effects of GPS position inaccuracies on their V2V intersection protocols were also studied by implementing realistic GPS model.

Kurt Dresner and Peter Stone proposed a novel intersection control mechanism called Autonomous Intersection Management (AIM), and in particular described a First Come, First Served (FCFS) policy to coordinate AVs through an intersection [5]. Their approach employs a reservation-based system by which cars request and receive time slots from the intersection during which they may pass. The approach studies the effects of different levels of granularity which is the way to divide intersection space. Higher granularity gives the intersection more flexibility to guide vehicles; however, the computational complexity increases proportional to the square of the granularity. Thus there always exists a tradeoff between computational complexity of the AIM system and flexibility for intersection.

After proposing several simple yet effective rules for V2V and Vehicle-to-Infrastructure (V2I) coordination, system-wide safety is established in [6]. The paper proposed a framework for intersection management that integrates decisions made by the intersection in the discrete domain for vehicle ordering with decisions made by each vehicle in the continuous domain for its safety-guaranteed motion. As stated in the paper, the proposed concurrent intersection crossing algorithm orders all vehicles according to arrival time to the intersection communication region, route priority or other properties of every vehicle. Based on the ordering, a vehicle with the highest priority will be permitted if its route is not conflicting with routes of other already permitted vehicles.

Manuscript received March 19, 2015; revised August 27, 2015.

Otherwise, the vehicle will prepare to stop at the intersection entrance. And the vehicle will be permitted once the permitted vehicles whose routes are conflicting with it have cleared the intersection. Then it will accelerate to cross the intersection from the permission moment.

This paper studies an approach to improve intersection efficiency with guaranteed safety. In the paper, a novel concept named *Discrete-Time Occupancies Trajectory (DTOT)* is proposed to coordinate AVs' motions inside an intersection. DTOT is the vehicle occupancies sequence inside an intersection ordered by time. The algorithm is only dealing with the first vehicle on each lane (*Head Vehicle*) which is similar to Reference [7], but the approaches to coordinate vehicles are totally different. If the AV is not the Head Vehicle, it will just follow other cars. In our DTOT-based intersection management scheme, unlike other approaches [4-5], vehicles can have different speeds when they enter the communication range of an intersection. It is assumed that an AV always wants to go through an intersection as quickly as possible. So AVs in our algorithm always try their best to reach maximum allowed speed within the intersection. Also different speed limit is set for different route since passenger comfort is considered. The advantages of our DTOT-based intersection management over other approaches are, (1). The approach is dealing with vehicles' actual occupancies inside an intersection which is different from the grid cells of most existing approaches, thus granularity issues do not need to be taken care of. (2). Different DTOTs represent different behavior of a vehicle inside an intersection. Hence, a vehicle has the flexibility to choose the way it wants to cross the intersection by sending corresponding DTOT. Also a new measure of intersection control algorithms, *Effective Average Trip Time* is proposed in the paper which can give us comprehensive information about the performance of an intersection control algorithm.

The rest of the paper is organized as follows: Section II introduces the main idea of our interaction protocol between vehicles and intersection. Detailed explanation of the concept DTOT is shown in Section III. In Section IV, concrete algorithm and some functions are introduced. The simulation setup and corresponding results can be found in Section V. Finally we give conclusion and future work in Section VI.

II. Interaction between Vehicles and Intersection

The intersection traffic we are studying is composed of purely AVs, i.e. all vehicles are driven automatically by computers and equipped with a wireless communication device. They are provided by accurate vehicles' positions and critical information of intersections and roads. The intersection traffic is controlled by *Intersection Control Agent (ICA)*, which processes messages from AVs and makes adjustments on their proposed DTOTs. We assume that communication performances are perfect resulting in no packet drops or transmission delays and a vehicle is always able to keep a safe minimum distance to its front vehicle based on their current velocities. Following are definitions of some terms that will be used in the paper.

Space-time Confliction: Two vehicles are space-time conflicting if their routes are conflicting not only in space but also in time.

Head Vehicle*:* A vehicle on a certain lane that there is no other vehicle in front of it or the vehicle in front of it has begun to cross the intersection.

Timed States Sequence (TSS): The states sequence ordered by time which will be occupied by the vehicle along its trajectory inside an intersection. TSS is predicted by the vehicle assuming that it will go through the intersection the way it wants to. $TSS \coloneqq \{(t_k, X_k)\}_{k=1}^{n}$ where $X_k = (x_k, y_k, \theta_k)^T$ consists of the coordinate and orientation of the vehicle at time t_k.

Confirmed Vehicle: A vehicle who has proposed and got a confirmed DTOT from ICA.

Vehicles who want to go through an intersection always maintain up-to-date information about its velocity, distance to the enter line of intersection and so on. Based on these information, an AV which is the Head Vehicle of its lane sends a REQUEST message to ICA in order to claim its pass request. The REQUEST message contains sender, receiver, data and message ID. Here correspondingly they are *Vehicle Identification Number (VIN)* (unique for every vehicle), ICA, *Vehicle Information (VI)* and *Message Sequence Number (MSN)*. VI is composed of *Vehicle Size (VS)* and TSS. VS is the actual length and width of the vehicle. MSN is only changed when a vehicle generates a new message to send, thus it is a monotonically increasing number.

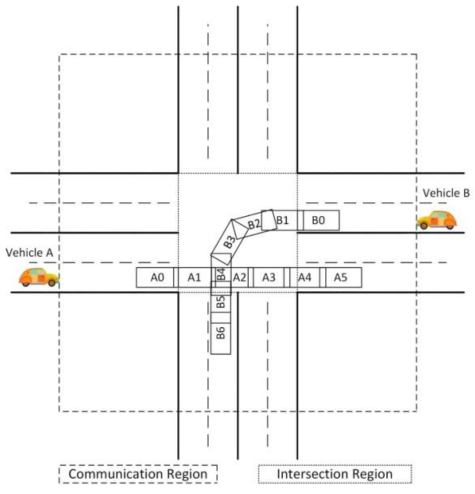

Figure 1. DTOTs of two conflicting vehicles

Based on received VI message, ICA expands the message into DTOT which are the ordered vehicle occupancies inside an intersection and corresponding time instants. An occupancy is the planned occupied space by the vehicle inside the intersection, and can be calculated by VI. DTOT gives ICA the vehicle's intended arrival lane, turning intention and other information related to the behavior of the vehicle inside the intersection. Fig. 1 shows an example situation when two Head Vehicles have conflicting DTOTs. Vehicle A has intersection occupancies {A0, A1, A2, A3, A4, A5} while vehicle B has {B0, B1, B2, B3, B4, B5, B6}. Notice that, for clear illustration, occupancies in Fig. 1 are very sparse. The number of occupancies in a DTOT in our simulation is much more than Fig. 1 to ensure accuracy. In the figure, *Communication Region* represents the range of wireless communication between vehicles and the intersection. *Intersection Region* is the shared area by all roads connected to the intersection. Detailed explanation of DTOT is in Section III.

Once ICA receives the REQUEST message, it will process and send RESPONSE message back to vehicle. Similar to REQUEST, this message also contains sender, receiver, data and message ID. Here they are ICA, VIN, TSS and MSN correspondingly. The ICA will check if the DTOT of an AV is space-time conflicting with other confirmed vehicles, if not, the field TSS in the RESPONSE is the same one proposed by an AV in its VI in the REQUEST; if yes, it will modify the DTOT to be not space-time conflicting with DTOTs of all other confirmed vehicles while the speed and acceleration inside the DTOT are achievable by the vehicle.

Once an AV receives RESPONSE, the AV is required to follow the occupancy trajectories provided by ICA.

III. Discrete-Time Occupancies Trajectory

In this section, we describe the meaning and properties of DTOT in detail. $DTOT_i$ stands for the Discrete-Time Occupancies Trajectory of vehicle i and is converted by ICA from TSS and VS of vehicle i. TSS starts with the state which the vehicle's front bumper first contacts the enter line (part of the boundary of the Intersection Region) of an intersection, and ends with state that the vehicle last touches the Intersection Region. Given a certain time interval, the number of elements in TSS depends on the vehicle's speed and route. ICA uses interpolation method to form DTOT consisting of connected and reasonable number of occupancies since we allow AVs to send DTOTs of unconnected or sparsely connected occupancies. To convert a vehicle's TSS to DTOT, we calculate the representation of rectangle from corresponding coordinate and orientation for every element in TSS. Since this rectangle represent the actual occupation of the vehicle inside the intersection, DTOT approach uses the intersection space more efficiently compared with other existing intelligent intersection management methods.

For an individual occupancy in a DTOT, we can obtain its *enter_time* and *exit_time* which correspond to the times when the vehicle first contact and totally out of the occupancy by taking the times of the previous and next occupancies which are the closest to the occupancy while having no overlapping area. For initial several occupancies in a DTOT which cannot find a previous occupancy that has no overlapping area with themselves, we take the first occupancy's time in the DTOT as these occupancies' *enter_time* since it makes no difference for the result of confliction check. Similarly, for last several occupancies in a DTOT, we take the last occupancy's time as their *exit_time*. Through this way of calculation, *enter_time* and *exit_time* of each occupancy can be arbitrarily precise, thus avoid the potential collisions from discretization. For example in Fig. 1, B0's time and B4's time will be the *enter_time* and *exit_time* of B2 respectively. In our simulation, we take enough number of occupancies for each vehicle to obtain accurate *enter_time* and *exit_time* for each occupancy.

Based on the rectangle representation of each occupancy, we can check if two occupancies are overlapping or not. The range [*enter_time*, *exit_time*] is the occupying time of one occupancy by a vehicle. Thus we can also check whether the two occupancies have conflicting occupying time or not.

Following terms are the important variables related to DTOT that will be used in our algorithm which is given in Section IV.

$FCO_{i,j}$*:* The first occupancy of vehicle i's DTOT which is space-time conflicting with *occupancies* of vehicle j's DTOT. If there is no conflicting occupancy, $FCO_{i,j} = \emptyset$.

CV_i*:* The set of vehicles whose DTOTs are space-time conflicting with vehicle i's DTOT. *Vehicles* are ordered by the times of corresponding FCOs in this set.

SV_t*:* The set of vehicles with confirmed DTOTs at time t. A confirmed DTOT has no confliction with all other vehicles' DTOTs that have already been confirmed. A confirmed vehicle will be deleted from SV_t once it exits the Intersection Region.

Now based on the definition of FCO we say that two vehicles' DTOTs is not conflicting with each other if $FCO_{i,j} = \emptyset$, and $FCO_{j,i} = \emptyset$; and space-time conflicting with each other if $FCO_{i,j} \neq \emptyset$, and $FCO_{j,i} \neq \emptyset$. In Fig. 1, the occupancies {A1, A2} and {B4, B5} of vehicles A and B have overlapping area. Then there are 4 cases for $FCO_{A,B}$ and $FCO_{B,A}$ depending on the earliest pair of occupancies that have conflicting occupying times. For example, if A1 and B4 have no overlapping occupying times but A1 and B5 have, then $FCO_{A,B} = \{\text{A1}\}$ and $FCO_{B,A} = \{\text{B5}\}$.

IV. Intersection Management Algorithm

In this section, the main intersection control algorithm which applies to ICA is described. The whole algorithm runs as follows: once ICA receives REQUEST from a Head Vehicle i, it first expands VI in the REQUEST message to DTOT. Based on the confirmed vehicles' DTOTs, ICA checks whether there are immediately front vehicles which will influence vehicle i's motion. If such vehicles exist, vehicle i's DTOT is delayed to avoid the influence of front vehicles. Then ICA checks whether vehicle i is conflicting with confirmed vehicles from other directions. If yes, ICA takes the first vehicle v in the conflicting vehicles set CV_i to update vehicle i's DTOT. Based on the updated DTOT, ICA obtains the conflicting vehicles set CV_i again. If CV_i is still not empty, ICA repeat previous steps until it becomes an empty set. Finally, ICA converts TSS from the final DTOT and sends it back to vehicle i for it to follow. Then ICA will continue to process another Head Vehicle's REQUEST.

There are several functions which are used in the main algorithm. Function Front Vehicle Check($DTOT_i$) checks whether vehicle i will be influenced by a front vehicle and delay vehicle i's DTOT if needed. As shown in Fig. 2, the front vehicle may be a vehicle which comes from other lane but has the same exit lane with vehicle i or has exactly same route with vehicle i. For vehicle i, if there is another confirmed vehicle which has same exit lane with vehicle i and will exit the intersection earlier, then they may collide immediately after crossing the intersection because of speed difference (when vehicle i has higher speed than the confirmed one). FrontVehicleCheck ($DTOT_i$) function will set this confirmed front vehicle's exit speed as vehicle i 's maximum allowed speed within intersection. For this situation, AIM approach [5] takes a simple method which gives 1s separation for the following vehicle. However, the constant separation time is not a 1-fit-all solution. The time should depend on the speeds of the two vehicles. Function Get_CV($DTOT_i$) returns the set CV_i consisting of vehicles who have space-time conflicting occupancies with vehicle i based on vehicle i's DTOT. The set CV_i only deals with the potential collisions inside the intersection. Function Update_DTOT ($DTOT_i, DTOT_j$) updates vehicle i's DTOT based on space-time conflicting occupancies between vehicles i and j. It will delay vehicle i to make it arrive at $FCO_{i,j}$ later than the exit time of vehicle j of $FCO_{j,i}$.

Algorithm 1 DTOT Intersection Control Algorithm

```
1: for vehicle $i \in$ unconfirmed Head Vehicle set do
2:
3:     Convert $VI_i$ to $DTOT_i$
4:
5:     $DTOT_i \leftarrow$ FrontVehicleCheck($DTOT_i$)
6:
7:     $CV_i \leftarrow$ Get_CV($DTOT_i$)
8:
9:     while $CV_i \neq \emptyset$ do
10:        $v \leftarrow$ Pop the first element in $CV_i$
11:        $DTOT_i \leftarrow$ Update_DTOT($DTOT_i, DTOT_v$)
12:        $CV_i \leftarrow$ Get_CV($DTOT_i$)
13:    end while
14:
15:    Store confirmed DTOT of vehicle $i$
16:    Send TSS to vehicle $i$
18:
17:end for
```

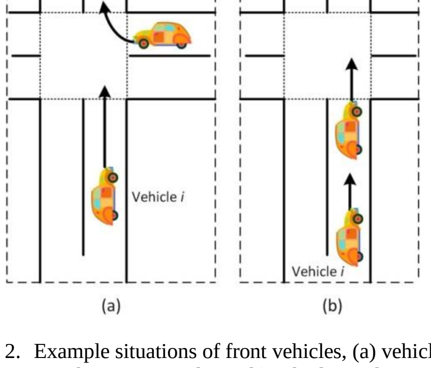

Figure 2. Example situations of front vehicles, (a) vehicles with different routes but same exit lane; (b) vehicles with same routes.

When a vehicle proposes its DTOT to ICA, we assume that it prefers to select the fastest way to pass the intersection which means the vehicle will try to use the maximum allowed speed to cross. To form VI for REQUEST message, the vehicle considers its current speed, distance to intersection and route it will use. After confirmed by ICA, if the vehicle's DTOT is modified, the times of occupancies inside intersection are always delayed based on current algorithm. Thus the vehicle can always meet the modified DTOT by decelerating to take longer time before entering the intersection. The worst case is that a vehicle needs to stop for some time before intersection to meet the given TSS from ICA.

V. Simulation

A. Simulation Setup

Traffic simulation is performed by the microscopic road traffic simulation package SUMO [12]. This simulator is widely used in the research community, which makes it easy to compare performance of different algorithms. Our intelligent intersection management algorithm is implemented by a Traffic Control Interface (TraCI) in SUMO.

The simulated scenario in our simulation is the traffic of a typical isolated four way intersection with 2 incoming lanes and 1 outgoing lane on each road. Communication Region is defined as $50\ m$ from the enter line of the intersection. $v_m = 70\ km/h$ is set as the maximum allowed speed for incoming roads. We generate vehicles with random velocity within the range of $40\% * v_m$ and v_m when they enter the Communication Region. To simulate intersection traffic as real as possible, we use different maximum allowed speeds for turning left, right and going straight. Although there are not specific speed limits for vehicles who are turning left or right, people are still using some lower speed to feel comfortable and maintain safety. In the paper, we choose conservative speed limits for turning based on experience from driving in daily life. We use $25\ km/h$ for right turning and $35\ km/h$ for left turning. For straight going vehicles, $65\ km/h$ is set as the speed limit.

The time step we are simulating is $0.05s$. The maximum acceleration and deceleration rate are $2\ m/s^2$ and $4.5\ m/s^2$. Vehicles have a size of 5 meters length and 1.8 meters width. We evaluate the performance of our algorithm in situations where vehicles are spawned randomly from each direction at different probabilities. In our simulation, we consider 3 different traffic volumes (we call 3 cases). Corresponding to the 3 cases, 1 vehicle is generated every 5, 2, and 1 second respectively. Traffic volume increases from case 1 to case 3 representing from light traffic to heavy traffic to see how the algorithm performs under different traffic situations. For each case, through different traffic generation time and randomly generated vehicles' routes, we test each case for twelve different traffic patterns. Average data of twelve different traffic patterns are used as the result for that case. Each simulation run is terminated when certain time limit (10 min) has been reached.

Fig. 3 shows a screen shot of simulation in SUMO when vehicles of different routes are within the intersection simultaneously without occurrence of collision. Inside the intersection, the straight going vehicle from East goes inside the intersection shortly after the vehicle from North to South clears the conflicting space. Vehicles whose DTOTs is not conflicting with these two can pass the intersection at the same time, for example the right-turning vehicle from South in the figure.

Compared with Concurrent algorithm [6], DTOT provides a more efficient way of coordinating vehicles to avoid unnecessary delay. Performance improvement has been validated through simulations of same traffic configurations of both algorithms. Simulation results of different cases are shown in Table I. To evaluate and compare the performance, we define several performance measures.

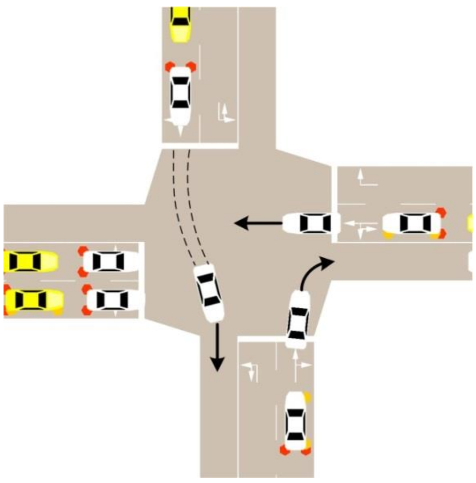

Figure 3. A screenshot of simulation which illustrates a situation when vehicles with conflicting routes cross the intersection simultaneously.

TABLE I. SIMULATION RESULTS COMPARISON.

		ρ	Crossed Vehicles			$\bar{\tau}_e$
			$\bar{\tau}$	σ_τ	ε	
Con	case 1	93.99%	13.98	8.58	51.74%	14.85
	case 2	60.24%	89.08	40.91	95.34%	150.77
	case 3	30.91%	125.09	48.33	97.17%	408.90
DTOT	case 1	95.11%	7.21	2.97	10.13%	7.57
	case 2	91.09%	20.47	15.54	55.86%	22.53
	case 3	57.09%	50.60	32.87	84.37%	88.92

Trip Time (τ): The difference between actual exit time of the intersection and the time the vehicle enters the intersection communication range.

Average Trip Time ($\bar{\tau}$): Average value of trip times of all crossed vehicles.

Throughput (ρ): The percentage of crossed vehicles against total generated vehicles.

Standard Deviation (σ_τ): This measure is computed based on the trip times of all crossed vehicles.

Stopped Percentage (ε): Obtained through dividing stopped vehicles number by crossed vehicles number.

Effective Average Trip Time ($\bar{\tau}_e$): The ratio of Average Trip Time to Throughput, that is $\bar{\tau}_e = \bar{\tau}/\rho$.

Since Average Trip Time is a measure only for those vehicles who have crossed the intersection which will not give us correct information about the algorithm, we introduce the measure Effective Average Trip Time which combines Throughput with Average Trip Time to capture the comprehensive performance of an algorithm.

Compared with concurrent algorithm, our algorithm increases Throughput and largely decreases Average Trip Time. As shown in the table, both algorithms have less and less Throughput with the increase of traffic volume. Also, larger decrease of Throughput is shown in concurrent algorithm compared with DTOT algorithm. DTOT's smaller values of Standard Deviation imply that DTOT is fairer than Concurrent algorithm. Stopped Percentage shows that less vehicles experience a stop at the intersection enter line for our DTOT algorithm thus saves energy. Effective Average Trip Time could tell us comprehensive information about the performance of intersection control algorithm. Efficiency and fairness of an algorithm are integrated in this measure. With more vehicles crossed and less Average Trip Time, DTOT algorithm has much less value of Effective Average Trip Time than that of concurrent algorithm.

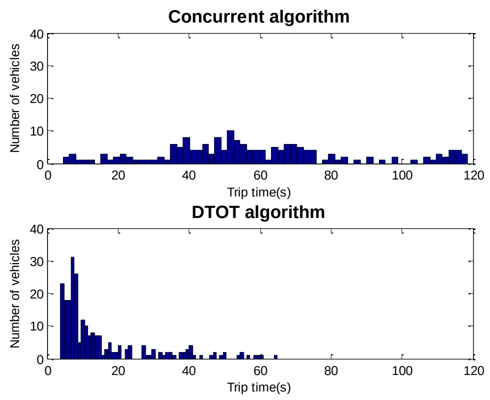

Figure 4. The histogram of trip times of crossed vehicles in a simulation

The histogram of Trip Times of crossed vehicles in one simulation run from case 1 and one particular traffic pattern is shown in Fig. 4. From the figure we can see that under the same traffic setting, for the crossed vehicles, DTOT algorithm results in less and concentrated trip times. On the other side, concurrent algorithm leads to much longer and wider distributed trip times. Compared with concurrent algorithm, DTOT is more fair and efficient for crossed vehicles.

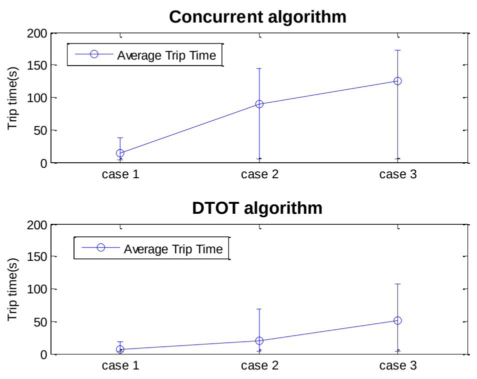

Figure 5. Comparison of trip times of 3 different cases between DTOT and concurrent algorithms.

Fig. 5 shows the maximum, average and minimum trip time for both algorithms under three different cases. With heavier traffic volume, both algorithms have large maximum trip times and average trip times. However, DTOT algorithm always performs better than concurrent algorithm in all three cases.

VI. Conclusion

We have developed an intelligent intersection control algorithm employing the concept DTOT. V2I interaction protocol has been established for interactions between vehicles and intersection. The paper introduces the concept of DTOT by which ICA is able to manage limited intersection space at a more accurate and efficient way. Compared with current existing AV intersection traffic managements, assumptions in the paper are more close to actual situations. Simulation results show that our algorithm achieves less Effective Average Trip Time compared with that of the concurrent intersection control algorithms in [6].

The proposed DTOT-based intersection management scheme is flexible enough so that it can handle unplanned intersection traffic situations. Currently, it is in-progress to enhance the algorithm to deal with sudden emergence of special vehicles such as emergency ambulance or police cars that have the highest priority in real traffic through efficient usage of intersection space. ICA may modify both occupancies' positions and times of a vehicle's DTOT in order to form a passage for special vehicles. Also, in the future, assumptions like perfect communication, accurate prediction of DTOT will be loosed to make the algorithm more applicable to real situation.

References

[1] H. Kowshik, D. Caveney, and P. Kumar, "Provable systemwide safety in intelligent intersections," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 3, pp. 804–818, 2011.

[2] J. Lee and B. Park, "Development and evaluation of a cooperative vehicle intersection control algorithm under the connected vehicles environment," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 1, pp. 81–90, 2012.

[3] J. Wu, A. Abbas-Turki, and A. El Moudni, "Cooperative driving: An ant colony system for autonomous intersection management," *Applied Intelligence*, vol. 37, no. 2, pp. 207–222, 2012.

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

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

[6] K. D. Kim and P. Kumar, "An mpc-based approach to provable systemwide safety and liveness of autonomous ground traffic," *IEEE Trans. Automat. Contr.*, vol. 59, no. 12, pp. 3341–3356, 2014.

[7] D. Carlino, S. D. Boyles, and P. Stone, "Auction-based autonomous intersection management," in *Proc. 16th IEEE Intelligent Transportation Systems Conference*, 2013.

[8] R. Horowitz and P. Varaiya, "Control design of an automated highway system," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 913–925, 2000.

[9] M. Campbell, M. Egerstedt, J. P. How, and R. M. Murray, "Autonomous driving in urban environments: approaches, lessons and challenges," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 368, no. 1928, pp. 4649–4672, Sep. 2010.

[10] Y. Kuwata, S. Karaman, J. Teo, E. Frazzoli, J. How, and G. Fiore, "Real-time motion planning with applications to autonomous urban driving," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 5, pp. 1105–1118, Sep. 2009.

[11] P. Varaiya, "Smart cars on smart roads: problems of control," *IEEE Transactions on Automatic Control*, vol. 38, no. 2, pp. 195–207, 1993.

[12] K. Daniel, E. Jakob, B. Michael, and B. Laura, "Recent development and applications of sumo - simulation of urban mobility," *International Journal on Advances in Systems and Measurements*, vol. 5, no. 3-4, pp. 128–138, December 2012.

Qiang Lu, currently is pursuing his Ph.D. in Electrical and Computer Engineering at University of Denver, received his Master's degree from Beihang University, China, in 2013 and Bachelor's degree from Southwest University, China, in 2010.
He is currently a research assistant in Dr. Kyoung-Dae Kim's group, with research interest in theories, simulations of intelligent intersection.

Kyoung-Dae Kim received the B.S. and M.S. degrees in mechanical engineering from Hanyang University, Seoul, Korea, in 1995 and in 1998, respectively, and the M.S. degree in computer science and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign, Urbana, IL, USA, in 2011.
He was a Postdoctoral Research Associate in the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX, USA. Currently, he is an Assistant Professor in the Department of Electrical and Computer Engineering, University of Denver, Denver, CO, USA. His research interest is developing theories, tools, and software frameworks to improve reliability and autonomy of cyber-physical systems, and their application to real systems such as smart ground and aerial transportation systems.