Distributed and Cooperative Fuzzy Controllers for Traffic Intersections Group

Jee-Hyong Lee and Hyung Lee-Kwang, *Member, IEEE*

***Abstract*—This paper presents a fuzzy traffic controller for a set of intersections and its simulation results. The controller of an intersection controls its own traffic and cooperates with its neighbors. It gets information from its traffic detectors and its neighbors. Using this information, the fuzzy rule base system gives optimal signals. It manages phase sequences and phase lengths adaptively to its neighbors' as well as its own traffic conditions. To carry out the performance evaluation of the controller, a simulator for intersections groups has been developed. The proposed method is compared with the vehicle actuated method which is one of the typical conventional methods. The average delay time of a vehicle is used as a performance index. The simulation results show good performances in the cases of time-varying traffic patterns and heavy traffic conditions.**

***Index Terms*—Cooperative systems, distributed control, fuzzy control, stimulation, traffic control.**

I. Introduction

THERE are many conventional methods for traffic signal control but they sometimes fail to deal efficiently with complex, time-varying traffic conditions. There has been some research done on traffic signal control based on fuzzy logic because fuzzy logic is adequate for the qualitative modeling of complex systems [1], [8], [9], [12], [16]–[18], [21]. However, most existing research based on fuzzy logic is devoted to simple traffic conditions such as those at a single intersection. Thus, those approaches are not suitable for the case of successively located traffic intersections. In the real world, especially in metropolitan areas, there are many intersections and they locate close to their neighbors, so it is not appropriate to independently consider an intersection from its neighbors [6], [19].

This paper deals with a set of intersections, i.e., an intersections group. We have developed a controller which changes both phase sequences and phase lengths of traffic signals adaptively to traffic conditions. For the performance evaluation of the developed controller, a simulator for intersections groups has also been developed.

In the next section, the basic theory of traffic signal control and related researches are summarized. The overview and details of the developed traffic controller are presented in Section III. Section IV shows the developed simulator and the simulation results. Finally, we make conclusions in the last section.

Manuscript received November 10, 1996; revised January 5, 1998 and July 22, 1998. This work was supported by KAIST Center of Artificial Intelligence Research.

The authors are with the Department of Computer Science, Korea Advanced Institute of Science and Technology (KAIST), Taejon 305-701, Korea.

Publisher Item Identifier S 1094-6977(99)02774-1.

II. Related Research

First, we briefly explain some basic terminologies: link, phase, and cycle. To explain them, it is assumed that vehicles keep to the right and traffic controllers manage only left-turning and straight-going traffic flows. Fig. 1(a) and (b) show a link and a four-phase cycle graphically. A *link* is a road connecting two intersections, a *cycle* is a turn of traffic signals and a *phase* is a traffic signal or the time duration that a signal continuously lasts [11]. A phase is represented by drawing the traffic flows which have the green signal. For example, Phase 1 in Fig. 1(b) is the phase for the traffic from east to west and from west to east.

Briefly speaking, controlling traffic signals is determining which phases are to be involved in a cycle and how long they should be. In the case of an ordinary four-way intersection, eight phases are possible as shown in Fig. 2. In order to control the traffic signals at a four-way intersection, we first have to choose the phases which construct a cycle. All eight phases are not required because some of them are sufficient to cover all traffic flows. For example, Phases 1–4 of Fig. 2 include four straight-going and four left-turning traffic flows.

The phases which form a cycle and their lengths should be carefully determined because they have a strong effect on the efficiency of traffic controllers. For example, if there is heavy traffic from east to west and from west to east, whereas other flows are less heavy, then we have to give a long green signal to the phases which include at least one of both heavy traffic directions. In that situation, if Phases 1–4 of Fig. 2 are selected, then only Phase 1 will have a long green signal because both heavy traffics belong to the phase. But if Phase 1 and 2 are replaced with Phases 5 and 6, both of them should have long green signals. It makes the cycle long and decreases the performance of the traffic controller. In the case where neighboring intersections are considered, we have to decide when each cycle starts to be in harmony with the neighbors [10]. If intersections are successively located, we have to coordinate an intersection with its neighbors to increase the performance of traffic controllers.

There are several research on traffic control using fuzzy theory. However, most of them controlled a single isolated intersection, not a set of intersections [2]–[5], [14]. The controllers proposed in those researches have from two to four inputs and generate one or two outputs. The output is the remaining time for the current green signal or the length

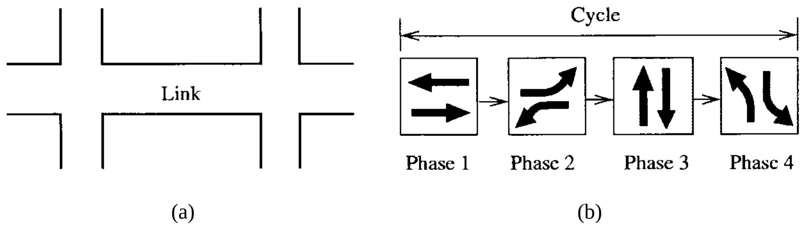

Fig. 1. (a) Link and (b) phase and cycle.

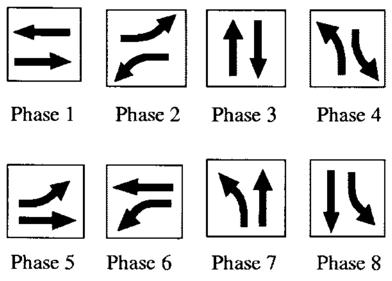

Fig. 2. Possible phases in an ordinary four-way intersection.

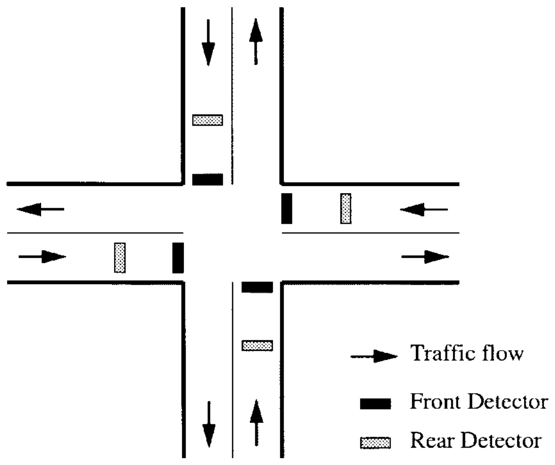

Fig. 3. Location of the detectors.

of the next green signal. Hoyer's controller generated one more output, the next phase, but it selected the next phase from a restricted phase set [4]. All the controllers fixed the sequence of phases and adapted only the length of the green phase except for Hoyer's.

Researches on controlling traffic of a set of intersection have also been performed [7], [13], [15], [20]. References [7], [13], and [20] were based on distributed fuzzy controllers which adapted to the traffic condition. However, they dealt with simple traffic conditions, such as, regularly located intersections [7], [20], inclusion of one way roads [13] and so on. References [13] and [20] did not change the sequence of phases. Changing sequence of phases may confuse the drivers, but in the view point of the performance of traffic controllers, it is needed. If the sequence of phases is fixed and traffic conditions are often changed, the traffic signal cannot effectively deal with the traffic conditions. Thus the performance of traffic controller may decrease. Reference [15] changed traffic signals by the reinforcement learning and genetic algorithms. It used a learning scheme converging only when the environment was stationary, and evaluated new parameters of traffic signals by applying them to intersections. For this reason, it might be difficult to effectively adapt traffic signals to varying traffic conditions.

III. Traffic Fuzzy Controller

In this section, we present the details of the proposed controller.

A. Overview

In order to control traffic intersections groups, we adopt a distributed scheme; that is, we locate a controller at each intersection and give the controller the whole control over the local traffic, i.e., the traffic of the corresponding intersection. It changes not only the phase lengths but also the phase sequences adaptively to the traffic situation. In addition, we let the controllers cooperate with their neighbors to achieve the control over the entire intersections group.

In order to develop a cooperation mechanism, two features are employed. One is *synchronizing traffic signals*. A controller tries to synchronize its traffic signals with its neighbors'. Such synchronization scheme aims to reduce the total delay time of waiting vehicles as well as providing drivers with convenience. The other feature is *controlling outgoing vehicles* in the case where a neighboring intersection has many vehicles. When an intersection is highly congested, the number of vehicles coming into the intersection should be reduced. If the number of incoming vehicles overruns the capacity of the intersection, the congestion would spread to its neighbors and finally all intersections would have traffic jam. For intersection cooperations, it is assumed that a controller can communicate with its neighbors and get information on their state through communication lines.

Vehicle detectors are important in the real-time control. It is assumed that there are two detectors to get local traffic information in a lane: the *Front Detector* and the *Rear Detector*. The Front Detector is located at the intersection and the Rear Detector is at a certain distance from the intersection. Fig. 3 shows the location of the detectors. We assume that a detector can count the number of vehicles passing through it. Basically the detectors are located according to the link length. In the case where the length is longer than or equal to 400 m, we locate the Rear Detector at 150 m from the intersection. In other cases, we reduce the distance between the two detectors proportionally to the link length. For example, in the case of

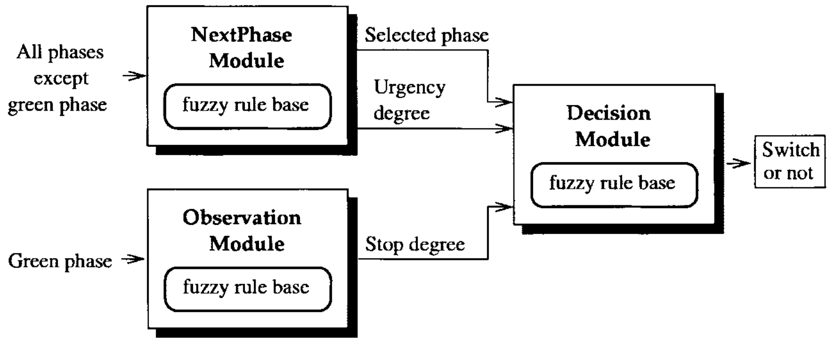

Fig. 4. Schematic diagram of the controller.

300 m link length, the Rear Detector is located at 112.5 m $(= 150 \times \frac{300}{400})$ from the intersection.

A controller gets two kinds of information: traffic information from its local detectors and from its neighbors' through communication lines. Based on those information, the controller changes phase lengths and phase sequences dynamically. Every 2 s, the controllers gather the two types of information and decide whether to switch the green phase or not.

For the decision of phase switching, the developed controller has three modules: the *NextPhase Module*, the *Observation Module*, and the *Decision Module*. The NextPhase Module has the role of selecting the most urgent phase among all the phases except the green phase. The Observation Module takes charge of observing the traffic condition of the green phase, and the Decision Module decides whether to switch the green phase according to the outputs of the first two modules. If the traffic condition of the phase selected by the NextPhase Module is more urgent than that of the green phase monitored by the Observation Module, the Decision Module will switch the green phase to the selected one. Otherwise the green phase continues. Each module has its own fuzzy rule base. Fig. 4 shows the schematic diagram of the controller.

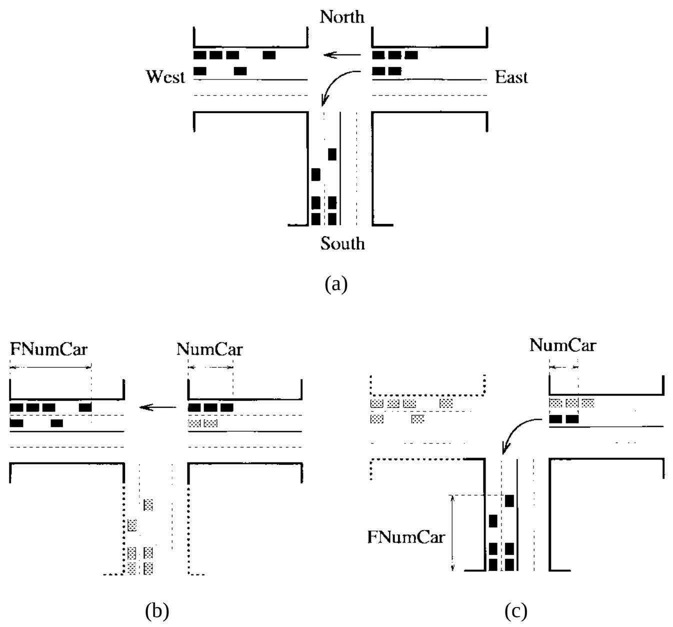

Fig. 5. *FNumCar* and *NumCar* for evaluation of the east traffic. (a) Traffic conditions, (b) *FNumCar* and *NumCar* of the traffic from east to west, and (c) *FNumCar* and *NumCar* of the traffic from east to south.

B. NextPhase Module

This module selects one candidate for the next green phase. It observes the traffic conditions of all phases except the green phase and selects the phase which is the most urgent among them. The inputs of this module are all phases except the current green phase and the outputs are the selected phase and its *urgency degree*. To select a phase, it generates the urgency degrees of the input phases. The *urgency degree of a phase* (UDP) means how bad the traffic condition of the phase is. This module compares the UDP's of all the input phases and selects the one which has the maximum UDP.

In order to get the UDP of a phase, this module evaluates the urgency degrees of all the traffic flows related with the phase. The *urgency degree of a traffic flow* (UDT) represents the traffic condition of the traffic flow. The UDP of a phase is the average of the UDT's of the traffic flows related with the phase. For example, Phase 6 of Fig. 2 has two traffic flows: one is from east to west and the other is from east to south. Thus, to get the UDP of Phase 6, first we get the UDT's of both traffic flows by applying their traffic conditions to fuzzy rules, and then defuzzify the fuzzy urgency values. The average of them becomes the UDP of Phase 6.

The fuzzy rules for evaluating the UDT of a traffic flow have four inputs *(NumCar, RedTime, SyncTime,* and *FNumCar)* and one output *(Urgency)*. *NumCar* is the number of vehicles waiting between the two detectors in a lane, and *RedTime* stands for the time duration that a traffic flow stays on red signal since the end of the last green signal for the traffic flow. Those two variables reflect the local traffic conditions. *SyncTime* is the remaining time until a vehicle having departed from an upstream intersection arrives at the intersection. *FNumCar* is the number of vehicles in the link between the intersection and the downstream intersection. The output, *Urgency*, is the UDT corresponding to the given traffic flow.

For example, let's consider the UDP of Phase 6 of Fig. 2. Fig. 5 shows *FNumCar* and *NumCar* of this phase. Fig. 5(a) is the traffic conditions of the phase, and Fig. 5(b) and (c) show *NumCar* and *FNumcar* of the two traffic flows. It is assumed

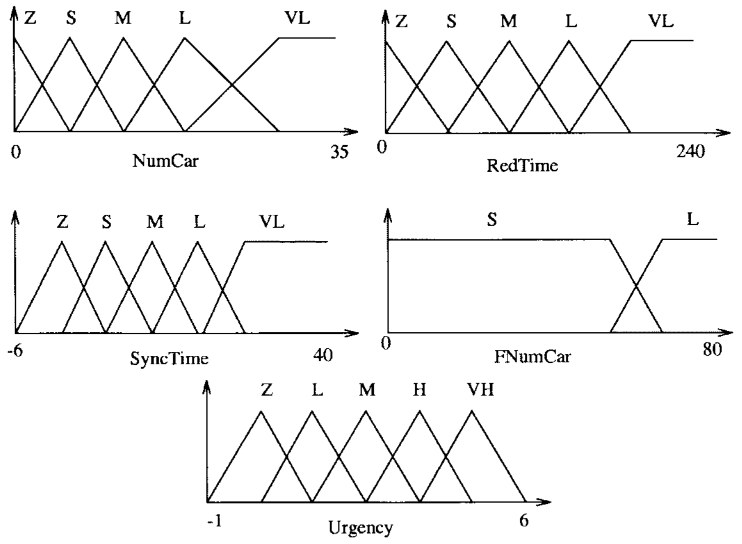

Fig. 6. The fuzzy sets of *NumCar, RedTime, SyncTime, FNumCar,* and *Urgency.*

that the outside lane is for the straight-going traffic and the inside is for the left-turning. Let us evaluate the UDT of each traffic flow. For the traffic from east to west, we get the number of vehicles waiting in the straight-going lanes of the east link (*NumCar*), the time duration that the straight-going traffic stays on red signal (*RedTime*), the remaining time for vehicles from its east neighboring intersection to arrive (*SyncTime*), and the number of waiting vehicles in the link between the intersection and its west intersection (*FNumCar*). Then, we apply those values to the fuzzy rules of the NextPhase Module and get the UDT of the straight-going traffic flow. The same process is also applied to the traffic from east to south. But, in this case, *NumCar* is the number of vehicles waiting in the left-turn lanes and *RedTime* is the time duration that the left-turning traffic stays on red signal. *FNumCar* is the number of vehicles in the link between the intersection and its south intersection.

The fuzzy rules of the NextPhase Module are generated so that UDT increases proportionally to *NumCar* and *RedTime.* As the number of waiting vehicles increases and/or the red signal lasts longer, the traffic condition is considered to become more urgent. For signal synchronizations, when vehicles having departed from the neighboring intersection arrive, the UDT of the arriving traffic flow will increase. However, if *FNumCar* becomes large, UDT should decrease because a large value of *FNumCar* means that there are too many vehicles in the next intersection. Thus the number of vehicles entering the next intersection should be reduced.

Table I shows some of this module's 36 fuzzy rules. For example, R_1 means that `if` *FNumCar* `is` **L**(Large) `then` *Urgency* `is` **Z**(Zero). That is, if the next intersection has many incoming vehicles then decrease UDT. R_5 says that `if` *NumCar* `is` **M**(Medium) `and` *SyncTime* `is` **L**(Long) `and` *FNumCar* `is` **S**(Small) `then` *Urgency* `is` **H**(High). That is, if the number of waiting vehicles is medium and it will take a long time for the vehicles from the neighboring intersection to arrive and the next intersection has a small number of incoming vehicles, then UDT becomes high in order to make waiting vehicles leave the intersection. Fuzzy sets of input and output variables are shown in Fig. 6.

TABLE I
SOME RULES OF THE NEXTPHASE MODULE

	NumCar	*RedTime*	*SyncTime*	*FNumCar*	*Urgency*
R_1				**L**	**Z**
R_2	**VL**	**L**		**S**	**VH**
R_3	**L**	**M**		**S**	**H**
R_4	**M**	**L**		**S**	**H**
R_5	**M**		**L**	**S**	**H**
R_6	**S**		**S**	**S**	**M**
R_7	**Z**		**Z**	**S**	**M**
...	...	...	...	...	...

C. Observation Module

The Observation Module observes the traffic conditions of the green phase. According to the result of the observation, it produces the *stop degree.* The stop degree indicates the possibility that the controller should stop the green phase.

The fuzzy rule base of this module takes *OutRate, RNumCar,* and *FNumCar* as its inputs and generates *Stop* as an output. *OutRate* is the number of outgoing vehicles per lane for the last five seconds. *RNumCar* is the number of vehicles remaining between the Front and the Rear Detector. Those two give the information on the current usage rate and the congestion degree of the phase. *FNumCar* is the same as *FNumCar* of the NextPhase Module. If the green phase involves more than one traffic flow, the maximum *OutRate, RNumCar,* and *FNumCar* of the traffic flows will be applied. For example, let us suppose that the current green phase is Phase 6 of Fig. 2. $OutRate_s$, $RNumCar_s$, and $FNumCar_s$ are for the traffic from east to west, and $OutRate_l$, $RNumCar_l$, and $FNumCar_l$ are for the traffic from east to south. Then the inputs for evaluating the stop degree of Phase 6 are the maximum of $OutRate_s$ and $OutRate_l$, the maximum of

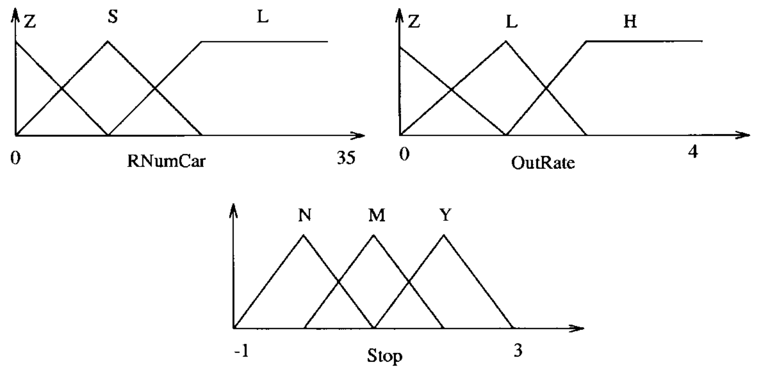

Fig. 7. The fuzzy sets of *RNumCar*, *OutRate*, and *Stop*.

TABLE II
SOME RULES OF THE OBSERVATION MODULE

	RNumCar	*OutRate*	*FNumCar*	*Stop*
R_1			**M**	**Y**
R_2	**Z**	**Z**	**S**	**Y**
R_3	**S**	**H**	**S**	**M**
R_4	**M**	**H**	**S**	**N**
...	...	...	...	...

TABLE III
SOME RULES OF THE DECISION MODULE

	Stop	*Urgency*	*Decision*
R_1	**N**	**H**	**N**
R_2	**M**	**M**	**N**
R_3	**M**	**H**	**Y**
R_4	**Y**	**M**	**Y**
...	...	...	...

RNumCar$_s$ and *RNumCar*$_l$, and the maximum of *FNumCar*$_s$ and *FNumCar*$_l$. We defuzzify the result of the fuzzy inference and use the value as the stop degree of the phase.

This module has ten rules. Table II shows some rules of the Observation Module. If a traffic flow has remained long enough on the green signal, the number of outgoing vehicles during the unit time *(OutRate)* and/or the number of remaining vehicles *(RNumCar)* will become small. Thus, if *OutRate* or *RNumCar* is small, the stop degree will be increased. If there are many vehicles at the next intersection, we make the stop degree higher to stop the vehicles entering the next intersection. For example, R_2 refers to the case where there is a small number of remaining vehicles between the detectors and the outgoing rate is low. In a statement form, R_2 can be written like this; `if` *RNumCar* `is` **Z**(Zero) `and` *OutRate* `is` **Z**(Zero) `and` *FNumCar* `is` **S**(Small) `then` *Stop* `is` **Y**(Yes). R_3 is that `if` $RNumCar$ `is` **S**(Small) `and` *OutRate* `is` **H**(High) `and` *FNumCar* `is` **S**(Small) `then` *Stop* `is` **M**(Maybe). In this case, the congestion is nearly removed but the road usage rate is high, so we cannot definitely say Yes or No. Fig. 7 shows the fuzzy sets of *RNumCar, OutRate,* and *Stop.*

D. Decision Module

The Decision Module makes decision whether to switch the green phase. Its inputs are *Candidate, Urgency*, and *Stop*, and its output is *Decision*. *Candidate* is the phase selected by the NextPhase Module and *Urgency* is its urgency degree (UDP). *Stop* is the stop degree of the green phase from the Observation Module. *Decision* is the decision on phase switching to *Candidate*. The result of fuzzy inference is a fuzzy set. This module defuzzifies it and switches the current green phase to the candidate if the defuzzified value is greater than the specified threshold.

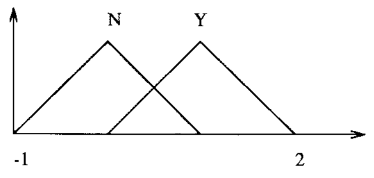

Fig. 8. The fuzzy sets of *Decision*.

This module will stop the green phase and give a green signal to *Candidate* if the urgency degree of the candidate or the stop degree of the current green phase is high. Table III contains some rules selected from the 15 rules of this module. R_1 says that although the candidate phase is congested *(Urgency* is **H**), if the stop degree is low *(Stop* is **N**) then keep the green phase *(Decision* is **N**, i.e., No changes). R_3 is the statement that `if` *Stop* `is` **M**(Maybe) `and` *Urgency* `is` **H**(High) `then` *Decision* `is` **Y**(Yes). The fuzzy sets of *Urgency* and *Stop* are the same as *Urgency* of the NextPhase Module and *Stop* of the Observation Module respectively. Fig. 8 shows the fuzzy sets of *Decision*.

IV. SIMULATION RESULT

The performance of the developed controller is evaluated by simulation. Eighteen traffic conditions are simulated and the results are compared with the vehicle actuated method [11]. We will describe the developed simulator and the vehicle actuated method in detail, and present the simulation results.

A. Simulator

The crux of a simulator is the generation of a realistic traffic flow. In the real situation, there are so many factors affecting a traffic flow that it is very difficult to consider all of them. We try to mimic the real vehicle motion by keeping *headways* (i.e., the time intervals between successive vehicles passing through a fixed point) similar to real ones.

The headway is a factor determining the saturation flow rate and the capacity of an intersection is determined by the saturation flow rate [11]. Therefore, if the values of headway are close to real ones, the capacity of an intersection and the motion of a vehicle in the simulator can be similar to those in the real situation.

In order to generate traffic flows, the following *traffic equations* were tried [3]

$$a(t+\tau) = \frac{4v(t)(v(t)-v_f(t))}{(x(t)-x_f(t))^2} + 0.2(v_{\text{limit}} - v(t)) \quad (1)$$

$$v(t) = v(t-\Delta t) + a(t)\Delta t \quad (2)$$

$$x(t) = x(t-\Delta t) + v(t-\Delta t)\Delta t + \frac{1}{2}\, a(t)\Delta t^2. \quad (3)$$

They are time-delay differential equations based on traffic flow theory and the last two are basic physics equations on the velocity and the position of an object. In the equations, $a(t)$, $v(t)$, and $x(t)$ are the acceleration, the velocity and the position of the vehicle at time t respectively. $v_f(t)$ and $x_f(t)$ are the velocity and the position of the leading vehicle (the vehicle immediately in front of it) at time t. v_{limit} is the speed limit of vehicles and τ is the reaction time of drivers.

The first term of (1), $4v(t)(v(t)-v_f(t))/(x(t)-x_f(t))^2$, can be regarded as the term for deceleration, and the second term, $0.2(v_{\text{limit}} - v(t))$, as for acceleration. That is, all drivers want to increase the speeds of their vehicles at the ratio of $0.2(v_{\text{limit}} - v(t))$, but at the same time, they will decelerate as the leading car become closer. But, there is a potential problem: if $v(t) > v_f(t)$, the first term will work as an acceleration term. While simulating with the equations, we found that the problem really occurred. After several tens of seconds from the start of the green signal, the third waiting vehicle becomes faster than the second and passes it ahead.

So, the equation (1) is modified as follows:

$$\lambda = \exp\left(\frac{-9.4(x_f(t)-x(t))}{v(t)^2}\right)$$

$$a(t+\tau) = 0.2\left(\frac{2}{1+\lambda} - 1\right)(v_{\text{limit}} - v(t)).$$

The equations are devised in a heuristic way, that is, a vehicle will accelerate if the distance between the vehicle and the leading vehicle is long, and will not accelerate if the leading vehicle is too close or the velocity is nearly equal to the speed limit. In addition to the equations, a heuristic rule is added for deceleration, such that, if the leading vehicle is too close, and the vehicle is faster than the leading vehicle, then make the velocity the same as that of the leading vehicle.

While simulating, 16.7 m/s is assigned to v_{limit} and a value between 0.9 and 1.5 s is randomly selected for τ. The headway values generated by the equations are compared with the values observed in the real traffic (Table IV). The headways in the table are measured in the following ways. It is assumed that 20 vehicles are waiting in a lane and the lane gets the green signal. Then, the first vehicle moves and then the following vehicle moves sequentially after τ. The time at which each vehicle goes out the lane is recorded. From the records, the nth headway value is obtained by subtracting the time of the $(n-1)$th vehicle from that of the nth. Thus, the accumulation from the first to the nth headway is the same as the time of the nth vehicle. We can see that the observed headways and the generated values are almost identical and thus the traffic equations are considered valid. Therefore, we will test and compare our proposed method with the vehicle actuated method by using traffic flows simulated by the above equations.

TABLE IV
HEADWAY COMPARISON

Sequence	Observed [10]		Simulated	
	HEADWAY	ACCUMULATION	HEADWAY	ACCUMULATION
1	2.4	2.4	3.0	3.0
2	2.0	4.4	1.9	4.9
3	2.0	6.4	1.8	6.7
4	1.9	8.3	1.7	8.4
5	1.8	10.1	1.6	10.0
6	1.7	11.8	1.7	11.7
7	1.7	13.5	1.7	13.3
8	1.8	15.3	1.4	14.9
9	1.7	17.0	1.4	16.5
10	1.6	18.6	1.5	18.0
11	1.6	20.2	1.6	19.6
12	1.6	21.8	1.6	21.2
13	1.6	23.4	1.5	22.7
14	1.6	25.0	1.6	24.3
15	1.7	26.7	1.5	25.8
16	1.6	28.3	1.5	27.3
17	1.6	29.9	1.6	28.9
18	1.7	31.6	1.5	30.4
19	1.5	33.1	1.5	31.9
20	1.6	34.7	1.5	33.4

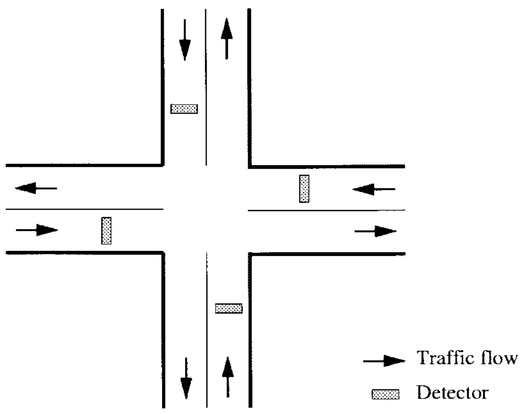

Fig. 9. Detector location of the vehicle actuated method.

B. Vehicle Actuated Method

In order to compare the proposed method with existing methods, the vehicle actuated method [11] is used for com-

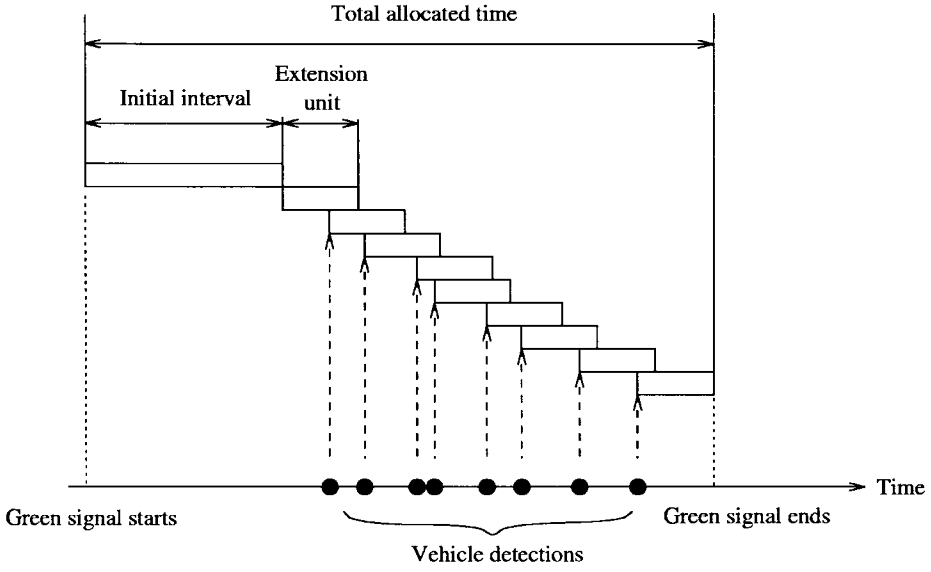

Fig. 10. Schematic diagram of time allocation of the vehicle actuated method.

parison. It is chosen because it is one of the typical methods, which uses vehicle detectors and changes the phase length like the proposed method.

It is assumed that the vehicle detectors are located as in Fig. 9. This method uses three parameters: the *initial interval*, the *extension unit time*, and the *extension limit*. If a phase takes the turn for the green signal, it gets the green for an initial interval. After the initial interval elapses, the green signal is extended by an extension unit time. If a vehicle is detected during the extended time, the green signal is extended once more. But the green will not be extended any more if no vehicle has been detected during the last extended time or if the total length reaches the extension limit. Fig. 10 graphically shows the time allocation. The dots indicate vehicle detections, and the dotted arrows point to the starts of extensions. In the figure, during the first extension a vehicle is detected, so the signal is extended right after the detection.

It is assumed that a detector is located at 45 m before an intersection, so seven or eight vehicles (passenger cars) can wait between the detector and the intersection. The initial interval should be long enough for the vehicles waiting inside the detector to drive through the intersection. According to Table IV, it takes 15 or 16 s for the eighth waiting vehicle and thus the initial interval is set to 16 s. Since it is assumed that detectors are at 45 m before an intersection and the speed limit of the vehicles is 16.7 m/s, the extension unit time is set to 3 s ($45/16.7 \approx 3$). A detected vehicle should be able to arrive at the intersection in an extension unit time. The limit of total length is set to 60 s.

C. Results

The proposed method and the vehicle actuated method are simulated under the same conditions. Simulations were performed under 18 traffic conditions: three *intersections groups* and six *traffic plans*. Three intersections groups have seven, nine, and 13 intersections, respectively. They are called

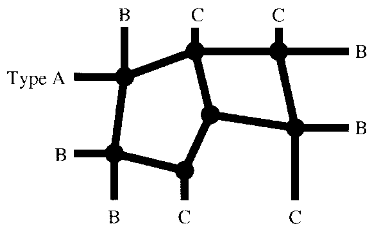

Fig. 11. Intersections—Group*1* with seven Intersections.

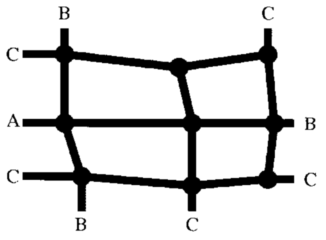

Fig. 12. Intersections—Group*2* with nine Intersections.

Intersections-Group*1*, Intersections-Group*2*, and Intersections-Group*3*. Figs. 11–13 show their rough maps. All links are two-way and have eight lanes (four lanes for one way). We assume that only passenger cars exist and there is no crosswalk. The intersections groups consist of two kinds of intersections: intersections with three corners and with four corners. In the case of intersections with four corners, the first lane from the center of a road is occupied by vehicles to turn left, the second and the third lanes to go straight, and the fourth lane to turn right. The vehicles in the fourth lane can turn right on any signals. The left-turning and the right-turning traffic are each 20% of the traffic of a link. In the case of intersections with three corners, the first and the second lanes are for left-turns, and the others are for right-turns. The traffic volume of both directions are the same.

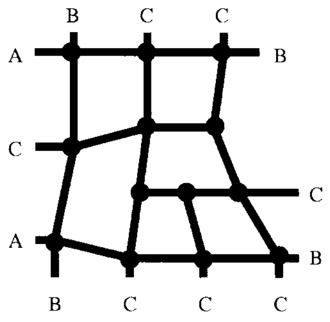

Fig. 13. Intersection—Group 3 with 13 Intersections.

TABLE V
TRAFFIC GENERATION PLANS

Generation Plan Number		Traffic Generation Rate(vehicle/h)		
		Type A	Type B	Type C
Steady Traffic	Plan 1	1700	1600	1500
	Plan 2	1900	1800	1700
	Plan 3	2100	2000	1900
Time-Varying Traffic	Plan 4	1500 → 2100	1400 → 2000	1300 → 1900
	Plan 5	1700 → 2100	1600 → 2000	1500 → 1900
	Plan 6	1900 → 2100	1800 → 2000	1700 → 1900

We call the links connected to only one intersection the *input-links*. The input-links are divided into 3 types according to their input traffic flow rate: Type A, Type B, and Type C. Any traffic conditions of an intersections group is generated from the combination of these three types. For example, the Intersections-Group *1* in Fig. 11 includes one Type A link, five Type B's, and four Type C's.

In order to simulate under various traffic situations, the six traffic plans in Table V are applied to the three intersections groups. The plans consist of three steady traffic conditions (Plans 1–3) and three time-varying conditions (Plans 4–6). Plan 1 is for light traffic, plan 2 is for medium and plan 3 is for heavy. The number in a cell is the number of input vehicles per hour. For example, if Plan 1 is applied, 1700 vehicles/h will be generated into Type A links, 1600 into Type B, and 1500 into Type C. In time-varying traffic conditions, there are two numbers. When the simulation starts, vehicles are generated at the rate of the first number, and as time passes the rate smoothly increases up to the second.

The average delay time of a vehicle at an intersection is used as an index of performance. The simulation results are summarized in Tables VI–VIII. Each table shows the delay time measured in second, and the improvement of the proposed method over the vehicle actuated method.

The proposed method shows good performance in all cases. In steady traffic conditions, it shows improvements from 3.5% to 8.4% over the vehicle actuated method. In time-varying conditions, improvements from 4.3% to 13.5% were obtained. However, it is pointed out that the proposed method shows small improvement in steady and heavy traffic conditions (Plan 3). We think such results come from the fact that in those conditions, the traffic volume is close to the capacity of intersections and thus there is little room for improvements.

TABLE VI
SIMULATION RESULT OF INTERSECTIONS—GROUP 1

	Delay Time(sec)		Imprv.
	FUZZY	ACTUATED	(%)
Plan 1	52.7	57.2	7.9
Plan 2	65.7	70.8	7.2
Plan 3	99.6	104.2	4.4
Plan 4	60.2	69.6	13.5
Plan 5	63.7	70.8	10.0
Plan 6	71.4	79.3	10.0

TABLE VII
SIMULATION RESULT OF INTERSECTIONS—GROUPS 2

	Delay Time(sec)		Imprv.
	FUZZY	ACTUATED	(%)
Plan 1	45.2	49.2	8.1
Plan 2	61.6	66.8	7.8
Plan 3	90.2	93.5	3.5
Plan 4	61.6	64.5	4.5
Plan 5	59.4	65.4	9.1
Plan 6	66.2	73.0	9.3

TABLE VIII
SIMULATION RESULT OF INTERSECTIONS—GROUP 3

	Delay Time(sec)		Imprv.
	FUZZY	ACTUATED	(%)
Plan 1	54.4	59.2	8.1
Plan 2	65.4	71.4	8.4
Plan 3	100.2	105.5	5.0
Plan 4	60.8	68.8	11.6
Plan 5	64.2	72.6	11.6
Plan 6	72.6	75.9	4.3

V. CONCLUSION

We have proposed the traffic controller for intersections groups based on fuzzy logic and compared its performance with the vehicle actuated method by simulation. In order to control a set of intersections, the controllers manage traffic signals based on the traffic information which they gather. The controller gets information from its detectors as well as its neighbors. Using those informations, the fuzzy rule base system gives an optimal phase and a sequence of signals in the intersection. The controller not only manages its local traffic but also cooperates with its neighbors.

The results obtained from the simulations show the superiority of the proposed method over the vehicle actuated method in terms of average delay time. Since we made the controller of an intersection an active component, it can be applied to any situations independently of the number of intersections or the relative positions of intersections. The proposed method is being considered for commercialization by a company. In order to improve the performance in the case of steady and heavy traffic condition, other factors such as historical informations could be considered in the fuzzy inference system.

REFERENCES

[1] W. Pedrycz, *Fuzzy Control and Fuzzy Systems, second extended edition*. New York: Wiley, 1993.

[2] C. P. Pappis and E. H. Mamdani, "A fuzzy logic controller for a traffic junction," *IEEE Trans. Syst., Man, Cybern.*, vol. 7, pp. 707–717, 1977.

[3] M. Jamshidi, R. Kelsey, and K. Bisset, "Traffic fuzzy control: Software and hardware implementations," in *Proc. 5th Int. Fuzzy Systems Assoc.*, 1993, pp. 907–910.

[4] R. Hoyer and U. Jumar, "Fuzzy control of traffic lights," in *Proc. 3rd IEEE Int. Conf. Fuzzy Systems*, 1994, pp. 1526–1531.

[5] Y. G. Hong, S. W. Cho, and K. S. Choi, "A study on fuzzy traffic signal control," in *Proc. KFMS Spring Conf. '94*, 1994, pp. 238–243, (in Korean).

[6] H. Strobel, *Computer Controlled Urban Transportation*. New York: Wiley, 1982.

[7] J. H. Lee, K. M. Lee, and H. Lee-Kwang, "Traffic control of intersection group based on fuzzy logic," in *Proc. 6th Int. Fuzzy Systems Assoc. World Congr.*, 1995, pp. 465–468.

[8] C. C. Lee, "Fuzzy logic in control systems: Fuzzy logic controller—Part 1," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, pp. 404–418, 1990.

[9] ——, "Fuzzy logic in control systems: Fuzzy logic controller—Part 2," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, pp. 419–435, 1990.

[10] ——, *Interim and Final Report on Development of New Traffic Signal Control System*, Seoul Metropolitan Police, Seoul, Korea, 1991, (in Korean).

[11] C. W. Doh, *The principles of Transportation Engineering*. Seoul, Korea: Chong-Mun-Kak, 1989 (in Korean).

[12] H.-J. Zimmermann, *Fuzzy Set Theory and Its Applications*. Dordrecht, Holland: Kluwer-Nijhoff, 1985.

[13] G. Nakamiti and F. Gomide, "Fuzzy sets in distributed traffic control," in *Proc. 5th IEEE Int. Conf. Fuzzy Systems*, 1996, pp. 1617–1623.

[14] J. Favilla, A. Machion, and F. Gomide, "Fuzzy traffic control: Adaptive strategies," in *Proc. 2nd IEEE Int. Conf. Fuzzy Systems*, 1993, pp. 506–511.

[15] S. Mikami and Y. Kakazu, "Genetic reinforcement learning for cooperative traffic signal control," in *Proc. IEEE World Congr. Computational Intelligence*, 1994.

[16] C. B. Kim, K. A. Seong, J. O. Kim, Y. B. Lim, and H. Lee-Kwang, "A fuzzy approach to elevator group control system," *IEEE Trans. Syst., Man, Cybern.*, vol. 25, pp. 985–990, 1995.

[17] K. M. Lee, D. H. Kwak, and H. Lee-Kwang, "Fuzzy inference neural network for fuzzy model tuning," *IEEE Trans. Syst., Man, Cybern. B*, vol. 26, pp. 637–645, Aug. 1996.

[18] C. B. Kim, K. A. Seong, and H. Lee-Kwang, "Design and implementation of a fuzzy elevator group control system," *IEEE Trans. Syst., Man, Cybern. A*, vol. 28, pp. 277–287, May 1998.

[19] A. J. Al-Khalili, "Urban traffic control—A general approach," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-15, pp. 260–271, 1985.

[20] S. Chiu and S. Chand, "Self-organizing traffic control via fuzzy logic," in *Proc. 32nd IEEE Conf. Decision Control*, 1993, pp. 1987–1902.

[21] Y. D. Kim and H. Lee-Kwang, "High speed flexible fuzzy hardware for fuzzy information processing," *IEEE Trans. Syst., Man, Cybern. A*, vol. 27, pp. 45–56, Jan. 1997.

Jee-Hyong Lee received the B.S and M.S degrees in computer science from Korea Advanced Institute of Science and Technology (KAIST), Taejon, in 1993 and 1995, respectively. He is currently pursuing the Ph.D. degree in computer science at KAIST.

His current interest fields are fuzzy theory and applications, evolutionary computations and data mining.

Hyung Lee-Kwang (M'95) received the B.S degree from the Department of Industrial Engineering, Seoul National University, Seoul, Korea, in 1978, the M.S. degree from the Department of Industrial Engineering, Korea Advanced Institute of Science and Technology (KAIST), Taejon, in 1980, the D.E.A. and Dr.Ing. degrees from the Department of Computer Science, INSA de Lyon University, France, in 1982 and 1985, respectively, and the Dr.Etat degree from the Department of Computer Science, INSA de Lyon University, France, in 1988.

He was an Assistant Professor from 1985 to 1989, and is currently a Professor at KAIST. His research interests include fuzzy systems, artificial intelligence, Petri nets, and expert systems.