Ingeniería Investigación y Tecnología
volumen XX (número 2), abril-junio 2019 1-16
ISSN 2594-0732 FI-UNAM artículo arbitrado
Información del artículo: Recibido: 20 de julio de 2018, aceptado: 1 de febrero de 2019

http://dx.doi.org/10.22201/fi.25940732e.2019.20n2.020

Driving solutions for autonomous vehicles, and adaptive proposals to set the traffic lights times and to choose routes

Soluciones de conducción para vehículos autónomos, y propuestas adaptativas para establecer los tiempos de semáforos y elegir rutas

Carrillo-González José Gerardo
Consejo Nacional de Ciencia y Tecnología (CONACYT), México
Universidad Autónoma Metropolitana
Departamento de Sistemas de Información y Comunicaciones
E-mail: jgcarrilo@conacyt.mx
http://orcid.org/0000-0002-4147-6764

Abstract

Two objectives are pursued in this article: 1) with adaptive solutions, improve the traffic flow by setting the time cycle of traffic lights at intersections and reduce the travel time by selecting the vehicles route (treated as separated problems). 2) Avoid driving conflicts among autonomous vehicles (which have defined trajectories) and these with a non-autonomous vehicle (which follows a free path). The traffic lights times are set with formulas that continuously recalculate the times values according the number of vehicles on the intersecting streets. For selecting the vehicles route an algorithm was developed, this calculates different routes (connected streets that conform a solution from the origin to the destination) and selects a route with low density. The results of the article indicate that the adaptive solutions to set the traffic lights times and to select the vehicles path, present a greater traffic flow and a shorter travel time, respectively, than conventional solutions. To avoid collisions among autonomous vehicles which follow a linear path, an algorithm was developed, this was successfully tested in different scenarios through simulations, besides the algorithm allows the interaction of a vehicle manually controlled (circulating without restrictions) with the autonomous vehicles. The algorithm regulates the autonomous vehicles acceleration (deceleration) and assigns the right of way among these and with the human controlled vehicle.
Keywords: Adaptive control, algorithm, autonomous vehicles, route guidance, simulation.

Resumen

Se persiguen dos objetivos en este artículo: 1) Con soluciones adaptativas, mejorar el flujo de tráfico estableciendo los tiempos de los semáforos en intersecciones y reducir el tiempo de viaje seleccionando la ruta de los vehículos (dos problemas tratados independientemente). 2) Evitar conflictos entre vehículos autónomos (con una trayectoria definida) y entre estos con un vehículo no autónomo (que sigue una trayectoria libre). Los tiempos de los semáforos se establecen con fórmulas que continuamente recalculan los tiempos dependiendo de la cantidad de vehículos en cada calle de la intersección. Para seleccionar la ruta de los vehículos se desarrolló un algoritmo, este calcula diferentes rutas (calles conectadas que conforman una solución del origen al destino) y selecciona una ruta con baja densidad. Los resultados del artículo indican que las soluciones adaptativas para establecer los tiempos de los semáforos y seleccionar la ruta de los vehículos, presentan un mayor flujo de tráfico y un menor tiempo de viaje, respectivamente, que las soluciones convencionales. Para evitar colisiones entre vehículos autónomos que siguen una trayectoria lineal, se desarrolló un algoritmo, este se probó con éxito en diferentes escenarios mediante simulaciones, además el algoritmo permite la interacción de un vehículo controlado manualmente (circulando sin restricciones) con los vehículos autónomos. El algoritmo regula la aceleración (deceleración) de los vehículos autónomos y asigna la prioridad para pasar entre estos y con el vehículo controlado manualmente.
Descriptores: Algoritmo, control adaptativo, selección de rutas, simulación, vehículos autónomos.

Introduction

The necessity of a better control management at intersections (through traffic lights) and a suitable route guidance approach increases along with the vehicles increment on streets. The present study explores adaptive solutions to improve the traffic quality: an auto-regulated control at intersections, which sets the traffic lights times and has the purpose to relieve the streets with more vehicles, and a route choosing algorithm, with the intention to guide the vehicles through the streets with the least quantity of vehicles and reach the destination faster.

There is work related with adaptive traffic lights. In Gershenson & Rosenblueth (2012b) the traffic lights adjust its times according the vehicles demand, the streets are fragmented in cells and elementary cellular automata rules are applied, an algorithm with rules is implemented that basically counts the number of vehicles within a distance before the intersection, the street which count exceeds a threshold gets the green, also rules are implemented to prevent that streets with a bottleneck (occurring after the traffic light location) get the green, the self-organizing method shows better average speed and average flow, at different densities, than fixed traffic lights. In Gershenson (2004), with multi-agent simulations, three self-organizing methods are tested against traditional, the variables measured were: number of stopped cars, the average speed and average waiting times. The self-organizing methods implemented were: Sotl-request control, which counts vehicles to select the street with priority, Sotl-phase control, same as Sotl-request but with a minimum time imposed to terminate the green time, and Sotl-platoon control, in which platoons with no more than μ vehicles can pass before changing to red. These methods are self-organizing since the rules independently regulate each intersection, and despite that, global coordination is achieved. Self-organizing methods outperformed traditional control methods due its adaptive nature (instead of optimizing), also is concluded that the formation of platoons minimizes conflicts among vehicles. In De Gier *et al*. (2011) is shown the better performance of adaptive traffic lights against non-adaptive, the travel time mean and fluctuation is lesser in the adaptive case if the system is informed of the traffic situation in the upstream and downstream links, in contrast to have only the information of the upstream links. In Fouladvand *et al.* (2004) is proposed a traffic responsive signalization algorithm, following the concept of cut-off queue length and cut-off density, where is wanted to reduce the total delay of the intersecting streets, leading to the optimum signalization. The work in Gershenson & Rosenblueth (2012a) shows that using a self-organizing method the traffic lights control on cities can be improved for the drivers benefit. In Helbing *et al*. (2005) it is presented a fluid dynamic model to simulate traffic on roads with different lengths and capacities, the model is designed to easily simulate congested and free traffic, also throughputs and travel times are improved considering self-organization principles to set the interaction between vehicles and traffic lights. In Lämmer & Helbing (2008) is introduced an approach to coordinate incompatible traffic flow at intersections based on observed pedestrian flow activity (people walking), with optimization and stabilization rules, the self-organized control presents a low performance if the road neatwork is being used a lot, being necessary a proper combination to get a high performance (reduction of travel time and its variation). In Carrillo *et al*. (2018) is presented an algorithm that regulates the traffic at an intersection, without the need of traffic lights, keeping a balance in the number of vehicles passing from each direction, adequate for streets where no side has priority.

There is also work related between self-organization and traffic. The self-organization effects in the high and low density traffic flow phases are studied in Biham *et al*. (1992), where with a simple model the traffic flow is described. A self-organized traffic information system is proposed in Wischoff *et al*. (2003), where vehicles perform traffic analysis and the information is shared with other vehicles. In Viriyasitavat & Tonguz (2012) is proposed a management scheme that delegates the function of regulating the traffic to a leader vehicle, which serves as a virtual traffic light. With a set of rules, the priority of emergency vehicles is established, the simulations show that the emergency vehicles travel time is reduced, and the other vehicles travel time is not significant affected. In Tonguz *et al*. (2011) is presented a biologically inspired approach to solve transportation problems using the self-organizing paradigm.

The cause of driving accidents is usually due human mistakes, with autonomous vehicles (AVs) is intended to replace the human intervention, but some problems arises, as the design of a control to manage the AVs. The current study proposes a solution to achieve a safety driving interaction among AVs, and these with a human controlled vehicle. These findings contribute with knowledge to the intelligent traffic field. In Nagel *et al.* (2006), an algorithm that allows path planning and obstacle avoidance at 132 miles of unknown and rough territory is introduced. In Liao (2012), a modification of the A* algorithm in order to avoid collisions among unmanned aerial vehicles (UAVs) is presented. Wei & Dolan (2009) focus on autonomous driving, a prediction engine (for anticipating the intentions of the surrounding vehi-

cles), a cost function based scenario evaluation (that evaluates the predicted scenarios and generates strategies), a cost function based algorithm (used in driving ability, distance keeping, lane selecting and merge planning) and a freeway driving performance analysis (that combines qualitative and quantitative performance evaluations) were implemented. In Resende & Nashashibi (2010), two methods for real time trajectory planning, applied to automated driving and implemented for the HAVEit European project, are presented. First method considers the partial motion planning approach, and the second employs a fifth-degree polynomial to generate the trajectory (the coordinates to perform a lane change at constant speed). In Qian *et al*. (2014), is proposed that legacy vehicles (manually driven) respect car following rules to achieve the driving interactions between AVs and legacy vehicles.

Intersection control

The simulations conducted in this article were developed in the Unity Engine (Technologies, n.d.). The travel time (Yu-qin *et al*., 2013) to cross a road segment is slightly modified and presented in Equation 1.

$$T = t\left[1+\beta\left(\frac{D}{C}\right)^{n}\right] \tag{1}$$

The author exchange flow per density in Equation 1, from which $t = 40\ s$ is the travel time (in seconds) at free density traffic conditions, D = density (vehicles/m), C = density capacity (vehicles/m), which is set 1 vehicle each 5 meters, (equal in value to a flow capacity of 1 vehicle each 5 seconds, with different units), β = 0.15 and $n = 4$ are model parameters (Yu-qin *et al*., 2013). The vehicles speed is set with Equation 2.

$$v = l/T \tag{2}$$

where l is the road segment length. In this study a road segment (or street segment) starts and ends with an intersection, and each segment measures 1000 m. A vehicle's travel time (T) is set using Equation 1 with D as the number of vehicles circulating ahead, e.g. if there is a platoon of 3 vehicles approaching the segment, the travel time of the 1st vehicle arriving on the segment is calculated with D = 1, the 2nd vehicle with D = 2 and so on, therefore T modifies the vehicles speed through Equation 2. A vehicle's speed is recalculated (this apply for each vehicle on the segment) when a vehicle leaves or arrives at the segment, because the travel time depends of D and this variable is refreshed as the number of vehicles in front of the current vehicle.

The control logic is designed accounting for the number of vehicles on the segments, i.e. a segment with more vehicles, should has the priority. The meaning of the variables and constants employed are presented in Table 1. The letters *a* and *b* are used (interchangeable) for referring any two intersecting segments which traffic flow is related with traffic lights, in this case the south-north flow direction segment with east-west, and south-north with west-east. The capacity density on any segment is $C_a = C_b = C = 1$ *vehicle* / 5 *m* = 200 *vehicles* / 1000 *m*, as previous defined.

Table 1. Notation

C_a	Capacity density on segment *a*
C_b	Capacity density on segment *b*
D_a	Current density on segment *a*
D_b	Current density on segment *b*
t_{go}	Green time
t_{stop}	Red time
t_t	Total time of a traffic light cycle

Consider now P_{ab} (see Equation 3), which is the difference of the ratios D_a/C_a and D_b/C_b, from segments *a* and *b*, respectively. The ratio (of any segment) is in the range from 0 to 1, with 1 indicating that the segment is at the maximum capacity.

$$P_{ab} = \left(\frac{D_a}{C_a}\right) - \left(\frac{D_b}{C_b}\right) \tag{3}$$

If $P_{ab} > 0$, then on segment *a* the current density is higher than in segment *b*. The traffic lights times, given priority (larger green time) to segment *a*, are assigned with Equation 4.

$$t^a_{go} = \left(\frac{t_t}{2}\right) + \left(\frac{t_t}{2}\right) * P_{ab}, t^a_{stop} = t^b_{go}$$
$$t^b_{go} = t_t - t^a_{go},\ t^b_{stop} = t^a_{go}, \tag{4}$$

Otherwise, if $P_{ab} < 0$, the current density on segment *b* is greater than in segment *a*, then segment *b* has the priority. The times are assigned with Equation 5.

$$t^b_{go} = \left(\frac{t_t}{2}\right) + \left(\frac{t_t}{2}\right) * |\, P_{ab}\,|,\ t^b_{stop} = t^a_{go} \tag{5}$$

$$t^a_{go} = t_t - t^b_{go},\ t^a_{stop} = t^b_{go}$$

The case when $P_{ab} = 0$, means that the current density on segments a and b is the same, in this case

$t^a_{go} = t^b_{go} = t^a_{stop} = t^b_{stop} = \left(\frac{t_t}{2}\right)$. In the following $t_t = 20\ s$.

Simulations results

The first scenario to conduct simulations has two intersections, as presented in Figure 1, with three flow directions: down-up (south-north) in blue arrow, right-left (east-west) in red arrow and, left-right (west-east) in orange arrow.

Two simulations were performed using the scenario presented in Figure 1: adaptive control vs. conventional control. The conditions for both simulations were the same: the vehicles from south-north were generated each 1 s, from east-west each 2 s and west-east each 3 s. As mentioned before, the street segments length is 1000 m, and the traffic capacity (per segment) is $C = 200\ veh\ /\ 1000\ m$. The control points, to stop the vehicles if t_{stop} is enabled (red time), are located at 10 m before the intersections. When a vehicle finally stops braking, keeps 1 m with the vehicle in front, and after the vehicle in front starts to move, it waits 0.35 s to do the same. The time cycle of a traffic light is $t_t = 20$ s, the last 3 seconds of t_{go} are assigned to yellow time. From Equation 1, the travel time with $D = 0$ (with no vehicles in the segment) is $t = 40\ s$, then the max speed allowed is

$v_{\max} = \frac{1}{T} = \frac{1000m}{40s} = 25m/s = 90km/h$. For the conventional control, $t^a_{go} = t^a_{stop} = t_t/2$ and $t^b_{go} = t^b_{stop} = t_t/2$. The simulation time was 600 s, for the two simulations. Table 2 shows the number of vehicles passing the intersection(s), from each direction, during the simulation time. The vehicles from south-north direction, go through two intersections, from east-west and west-east directions, go through one intersection. The improvement was 47 vehicles comparing the adaptive control vs. conventional control.

A scenario with one intersection and four traveling directions (Figure 2) was implemented for the next simulations. Defining *Ts* as the time between vehicles arrivals, the vehicles from the south-north direction (in blue arrow) were generated each *Ts=0.5 s*, from north-south (yellow arrow) *Ts=1 s*, from east-west (red arrow) *Ts=3.5 s*, and from west-east (orange arrow) *Ts=4 s*, the simulation time was 300 s, other configurations remain the same.

Two simulations were performed, adaptive vs. conventional control. The number of vehicles passing the intersection is greater for the adaptive control (Table 3), with an improvement of 111 vehicles. The green time assigned to allow the flow from south-north and north-south directions (those with more vehicles circulating) in the conventional control is not enough, as the adaptive control simulation results suggest, with more vehi-

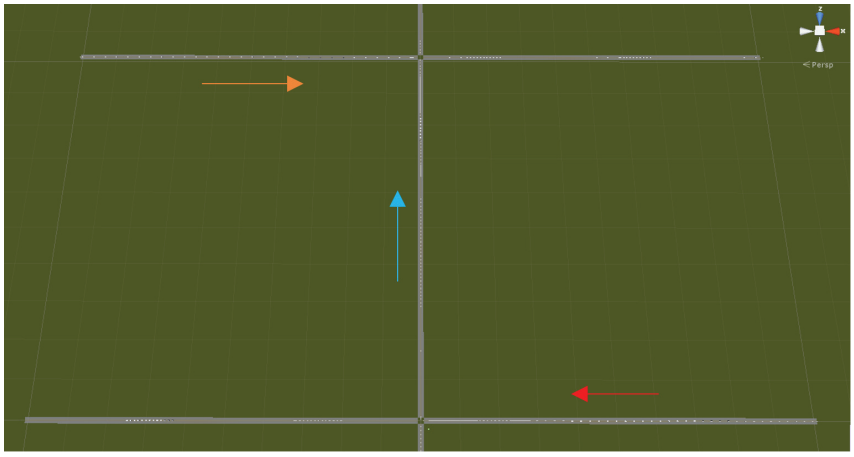

Figure 1. Two intersections scenario view

Table 2. Number of vehicles passing through the intersection(s)

	South-north	East-west	West-east	Total
Conventional	412 veh*	258 veh	172 veh	842 veh
Adaptive	462 veh	255 veh	172 veh	889 veh

*veh=vehicles

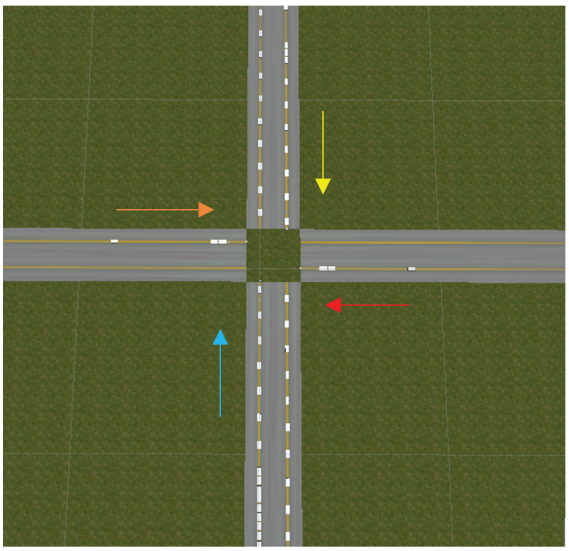

Figure 2. One intersection scenario view

cles passing the intersection from the directions mentioned before.

Route guidance

A solution to reduce the vehicles travel time is to distribute (with an adaptive paradigm) the vehicles along the streets, from the arriving to the destination place. The routing algorithm (annex A) was designed to establish a route with low density streets. The algorithm seeks for the streets that conform a route solution and with *density<threshold*. The threshold initial value is the lowest defined, if no solution is found, the threshold value is incremented, this process continues until a solution is found. For simplicity, the street network selected for simulating purposes has a grid form, with four blocks (Figure 3). If a vehicle's departure location is in the below-right corner and the destination location in the upper-left, with the horizontal streets with a flow direction from east-west and the vertical streets with flow from south-north, the route solution should involve to travel two segments to the left and two to the top, then there are six possible solutions as presented in Table 4, with u=up and l=left. The number of vehicles traveling on a street segment (Figure 3 shows the segment labels) is used to assign a weight to it, understanding that the more weight a segment has, it is harder to travel on it.

Simulation results

In this simulation the time cycle of the traffic lights (to regulate the traffic at intersections) is fixed (conventional control). A vehicle is created each 0.5 s in the origin, the intersection between $l_{1,1}$ and $u_{1,1}$. The first 120 vehicles follow pre-defined paths: these go through the $u_{1,1}$

Table 3. Number of vehicles passing the intersection

	South-north	North-south	East-west	West-east	Total
Conventional	201 veh	193 veh	60 veh	52 veh	506 veh
Adaptive	289 veh	213 veh	61 veh	54 veh	617 veh

Table 4. Path combinations

1	l	l	u	u
2	l	u	l	u
3	l	u	u	l
4	u	l	l	u
5	u	l	u	l
6	u	u	l	l

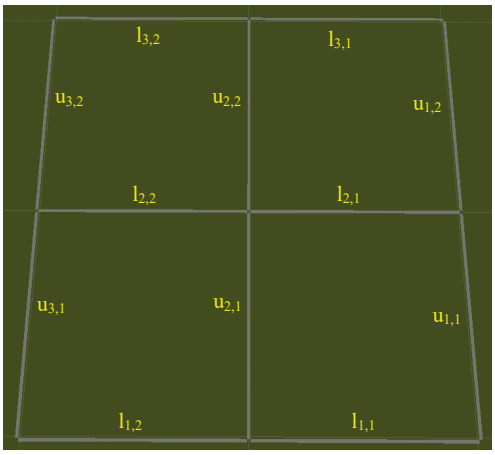

Figure 3. Street network

segment, then are distributed equally between $u_{1,2}$ and $l_{2,1}$, continuing with $l_{3,1}$ and $u_{2,2}$, respectively. The path of the vehicle 121 is defined with the routing algorithm. Figure 4 shows a two-dimensional plane, with units in meters. It can be seen in red the trajectory of the vehicle 120, which follows the defined path $u_{1,1}$, $u_{1,2}$, $l_{3,1}$, and $l_{3,2}$. The trajectory in blue is the followed by the vehicle 121, which path is $l_{1,1}$, $l_{1,2}$, $u_{3,1}$, and $u_{3,2}$, and was selected with the routing algorithm. Table 5 shows the vehicles entrance time (to $u_{1,1}$ and $l_{1,1}$ for vehicles 120 and 121, respectively) and the vehicles leaving time (by $l_{3,2}$ and $u_{3,2}$ for vehicles 120 and 121, respectively). In this simulation the travel time improvement of the vehicle guided (121) vs. not guided (120) is 14.86 s, this value is the difference between the travel times in Table 5. In this table, the entering and leaving time values represent the time in seconds (time = 0 when the simulation starts) when a vehicle enters and leaves the street network respectively. The travel time is the difference of the exit and entrance time. This result suggests that the routing algorithm effectively avoids streets with high density.

Autonomous vehicles interactions

In this section were performed simulations where the AVs interacts with each other without collisions, with

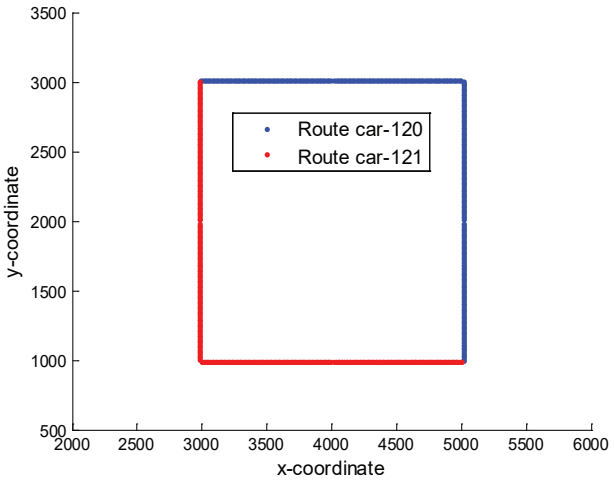

Figure 4. Vehicles trajectory

Table 5. Vehicles travel time

	Vehicle 120	Vehicle 121
Time when enters	61.52 s	62.03 s
Time when leaves	236.16 s	221.81 s
Travel time	174.64 s	159.78 s

this purpose an algorithm was proposed, in addition additional rules were implemented to allow the participation of a vehicle manually controlled. The braking algorithm (annex B) was employed to control the vehicles acceleration (deceleration) in the simulations from this section. The algorithm basically calculates the intersecting coordinates (if any) among the AVs trajectories, then selects the order for the AVs to accelerate or decelerate, avoiding conflicts. In the case where a manually controlled vehicle participates, the algorithm estimates the possible trajectory of this vehicle (based on the past coordinates), subsequently determines if the trajectories of the AVs and the manually controlled vehicle intersects, in that case regulates the acceleration (deceleration) of the AVs to allow the free-driving of the manually controlled vehicle.

Simulation 1

In this simulation, the AVs maximum speed (v_m) is 10 m/s, maximum acceleration a_m 2 m/s^2 and maximum deceleration d_m –2 m/s^2. Three AVs participates, which trajectories intersect in the same coordinate. As initial conditions, the AVs are stopped (Figure 5), the destination coordinates (or targets) are the points with the same color of the AVs, the black vehicle is the manually controlled vehicle (abbreviated CV and has no participation in this simulation). The AVs are numbered as next: red vehicle (or car) the 1st, orange vehicle the 2nd and, green vehicle the 3rd. When the simulation starts, the vehicles accelerate and follow its respective trajectory, later the 1st and 3rd vehicles decelerate because the 2nd vehicle has priority (assigned with the braking algorithm), see Figure 6.

After the 2nd vehicle has passed the conflict coordinate (where the vehicles trajectories intersect), the priority is assigned to the 1st vehicle, as can be seen in Figure 7. Finally, all the vehicles reach their target.

The variables of each vehicle involved in the simulation are presented in plots to explain how the braking algorithm operates. A vehicle without priority accelerates (if is the next in the priority queue) when the vehicle

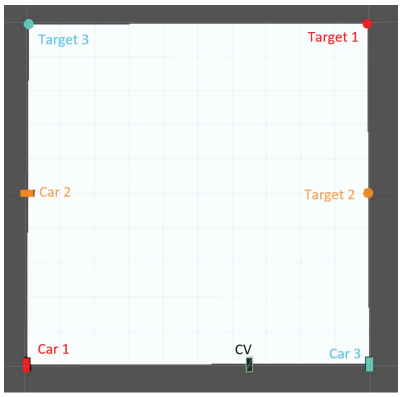

Figure 5. Simulation progress, part one

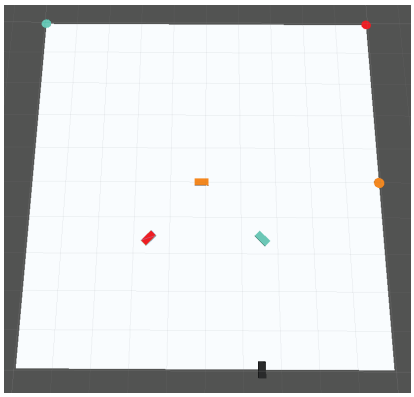

Figure 6. Simulation progress, part two

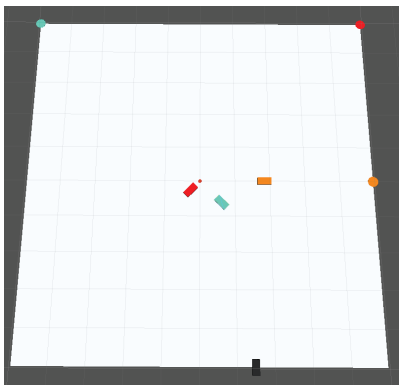

Figure 7. Simulation progress, part three

http://dx.doi.org/10.22201/fi.25940732e.2019.20n2.020

with priority has passed the conflict zone, i.e. the coordinate where their trajectories intersect. The plots that explain the simulation are presented next. In this simulation, $node_1$ is the intersection coordinate between the 1st and 2nd vehicles trajectories, $node_2$ is the intersection of the 1st and 3rd vehicles, and $node_3$ is the intersection between the 2nd and 3rd vehicles trajectories. In addition, the target of the 1st vehicle is called $target_1$, and so on for the other vehicles.

Figure 8 shows the variables evolution of the 1st vehicle: the 2nd vehicle is arriving first at $node_1$ than the 1st vehicle, this induces the braking of the 1st vehicle when the braking distance plus an aggregated distance (to stop with anticipation) becomes larger than the distance of the 1st vehicle with $node_1$, this happens in time approximately at 6.67 s. The braking distance = $(v^2 / -2d_m)$, with v = the current speed, and the aggregated distance = $sd = 5\ m$. The 1st vehicle accelerates when the 2nd passes $node_1$ approximately in time at 7.72 s, finally the 1st vehicle decelerates (to reach its destination) because the braking distance is greater than the distance between $target_1$ and the 1st vehicle, approximately at 14.44 s.

Figure 9 shows the variables evolution of the 2nd vehicle: the trajectory of the 2nd vehicle is not interrupted by any other vehicle, it can be observed that the acceleration is zero when its speed is at maximum (10 m/s), and only decelerates when the braking distance is larger than the distance with its target, at 10.06 s.

Figure 10 shows the variables evolution of the 3rd vehicle: this decelerates at 6.67 s because the 2nd vehicle has priority and the braking distance plus the aggregated distance is larger than the distance of the 3rd vehicle with $node_3$, eventually at 7.72 s the 2nd vehicle passes $node_3$. As the 1st vehicle has priority over the 3rd, the 1st vehicle passes $node_2$ at 9.80 s and enables the 3rd vehicle to accelerate. Finally, this decelerates to reach its destination at 15.99 s, because the braking distance is larger than the distance between the 3rd vehicle and $target_3$.

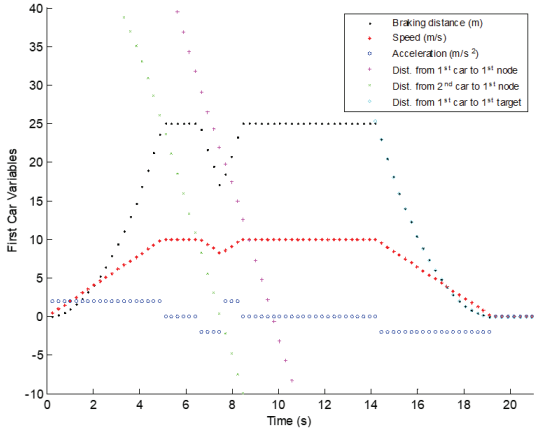

Figure 8. Time vs. 1st vehicle variables

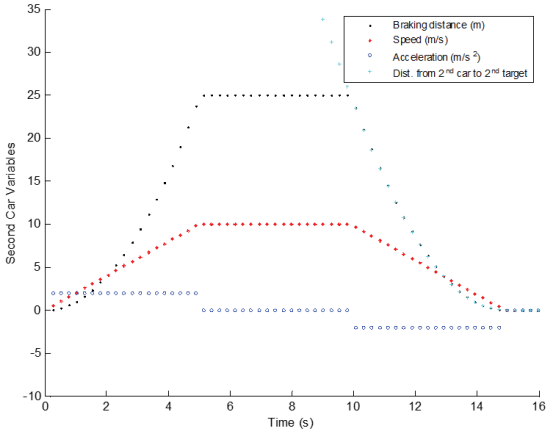

Figure 9. Time vs. 2nd vehicle variables

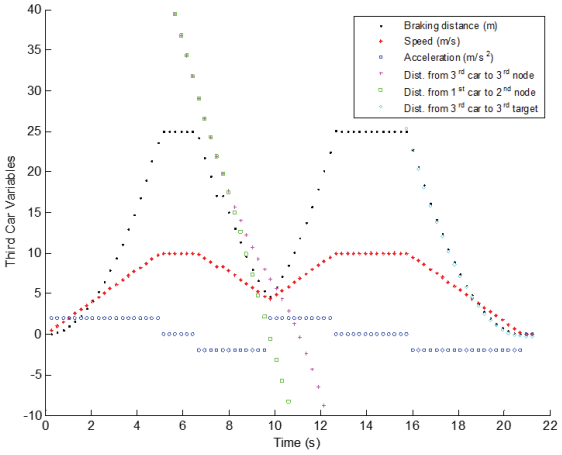

Figure 10. Time vs. 3rd vehicle variables

Simulation 2

In this simulation, the manually controlled vehicle participates. The Lagrange interpolation technique was considered for estimating the future coordinates of the CV, the formula is presented in Equation 6.

$$f(t) = \sum_{i=0}^{N} f(t_i) P_i(t) \tag{6}$$

where $f(t_i)$, from i…N, are the known function values (e.g. the two past and current position in the x-axis) and $f(t)$ is the value to be estimated (e.g. the future position in the x-axis). The Lagrange polynomial is calculated according Equation 7.

$$P_i(t) = \prod_{j=0, j\neq i}^{N} \frac{t - t_j}{t_i - t_j} \tag{7}$$

The time is the independent variable used for estimating the future CV coordinates (x^e, y^e), t_i is the respective time of $f(t_i)$ and t = current time + 1 s. With the predicted position (x^e, y^e) and the current position (x^a, y^a), the CV linear trajectory equation (slope and intercept) is calculated with Equation 8.

$$m^{cv} = (y^e - y^a) / (x^e - x^a)$$
$$b^{cv} = y^a - (x^a * m^{cv}) \tag{8}$$

The estimated intersecting coordinate $(x^s_{i_i}, y^s_i)$ of the CV with the i AV is calculated with Equation 9, with b^{av}_i and m^{av}_i are the slope and intercept, respectively, of the AV linear trajectory.

$$x^s_i = -(b^{av}_i - b^{cv}) / (m^{av}_i - m^{cv})$$
$$y^s_i = m^{av}_i x^s_i + b^{av}_i \tag{9}$$

The CV is manipulated with the keyboard using the direction keys, if the "up arrow" key is pressed, the vehicle accelerates continuously 2 m/s^2 until the button is released, with the "down arrow" key decelerates –2 m/s^2, the "right arrow" key rotates the vehicle by 0.8 degrees to the right and in the opposite direction if the "left arrow" key is pressed.

The initial vehicles position is presented in Figure 11. The trajectory followed by the CV (a straight line in the south-north direction) first causes the deceleration of the 3rd vehicle, later induces the braking of the 2nd vehicle and this causes the braking of all the AVs (Figure 12). A node is referred as the collision coordinate between the controlled vehicle and an autonomous vehicle: node$_1$ is the collision coordinate (CC) between the controlled vehicle and the 1st autonomous vehicle, node$_2$ is the CC between the CV and the 2nd AV, and so on. In this simulation the braking distance = $(v_m^2 / -2d_m) + sd$, with sd = 8 m.

Figure 13 shows the variables evolution of the 3rd vehicle: at 4.8 s the 3rd vehicle decelerates because two rules: 1) the braking distance is larger than the distance from the 3rd vehicle to node$_3$ and, 2) the required time to reach node$_3$ by the CV is lesser than by the 3rd vehicle. The 3rd vehicle decelerates until the CV passes node$_3$ (plus 5 m) at 7.59 s, also decelerates at 9.61 s because the CV causes a bottleneck. The 2nd vehicle stops (because the CV obstructs its trajectory), this induces the braking of the 1st and 3rd vehicles. The 3rd vehicle accelerates

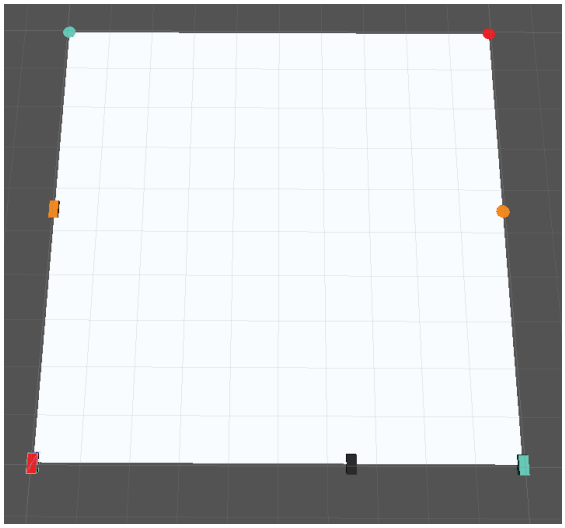

Figure 11. Simulation progress, part one

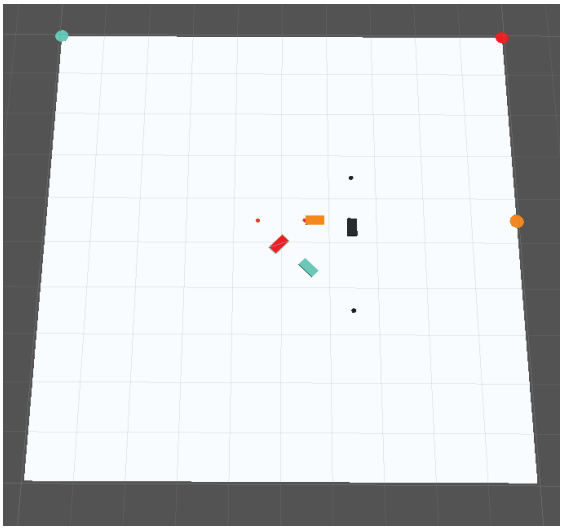

Figure 12. Simulation progress, part two

when the 1st vehicle is not obstructing its trajectory anymore and finally decelerates because the proximity of its destination.

Figure 14 shows the variables evolution of the 2nd vehicle: at 5.82 s the 2nd vehicle decelerates because the time to arrive to node$_2$ by the CV (3.19 s) is lesser than the required by the 2nd vehicle (3.79 s). Then at the 17.92 s, the CV passes node$_2$ over 5 m, allowing the 2nd vehicle to accelerate, finally at the 22.51 s the 2nd vehicle decelerates because it is arriving to its destination.

Figure 15 shows the variables evolution of the 1st vehicle: this decelerates at the 7.09 s due the bottleneck caused by the CV, at the 18.98 s the bottleneck ends and the 1st vehicle is able to accelerate, then at the 21.76 s decelerates because the CV intercepts its path. The deceleration rules are: the braking distance is larger than the distance from the 1st vehicle to node$_1$, the distance from the CV to node$_1$ is lesser than 5 m, and the time of the CV to reach node$_1$ is lesser than the required by the 1st vehicle. An advantage rule is always considered; the CV is 5 m closer (than actually is) and the AVs are 5 m away (than actually are) of the collision coordinate (or node), subsequently the time required by the CV to reach a node is zero when it is at least by 5 m (or closer) from the node. The 1st vehicle accelerates when the CV has passed node$_1$ by 5 m and later decelerates due the destination proximity.

Simulation 3

In this simulation the future position (1 s ahead of the current time) of the CV, using three previous position samples, is estimated. If it is detected that the trajectory (including the estimated future position) of the CV intersects the trajectory of an AV and a collision

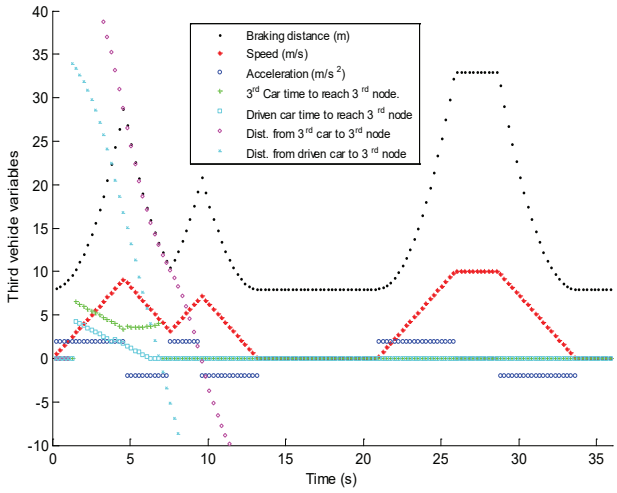

Figure 13. Time vs. 3rd vehicle variables

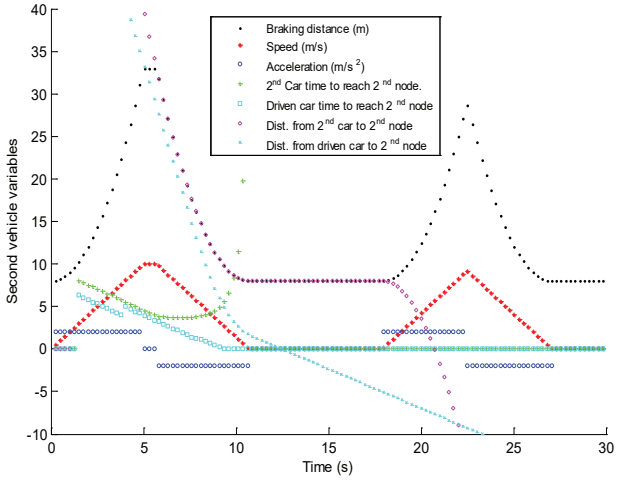

Figure 14. Time vs. 2nd vehicle variables

http://dx.doi.org/10.22201/fi.25940732e.2019.20n2.020

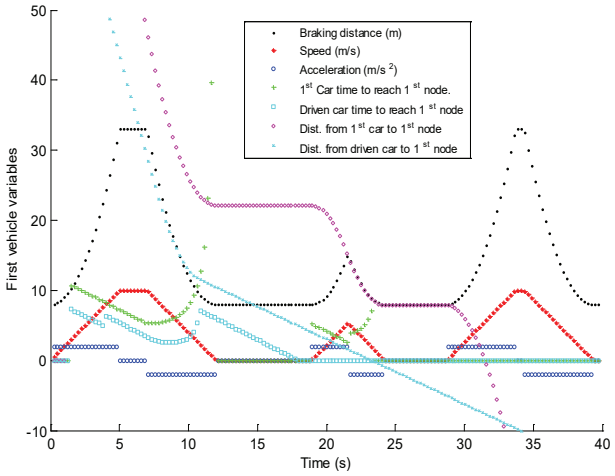

Figure 15. Time vs. 1st vehicle variables

will occur, measures are taken to decelerate (opportunely) the AV. Figure 16 shows: the estimated intersecting coordinate (IC), which is the collision coordinate between the CV and an AV considering the CV future position, and the current intersecting coordinate, which is the Collision Coordinate of the vehicles current linear trajectories.

Additional rules are introduced: if the AV is closeness in distance to the estimated intersecting coordinate, then it is concluded that the collision will occur in the IC, if not, will occur in the current intersecting coordinate. For the case of the collision occurring in the IC: if the CV can arrive (determined with the *calculation* function, in Annex B) at the collision coordinate in 5 s (or less) and the AV distance to arrive at the CC is lesser than the distance to brake = $(v_m^2/-2d_m) + sd$, with $sd = 12$ m, then the AV decelerates at –4 m/s^2. Figure 17 shows the trajectory (in a two-dimensional plane) of the controlled vehicle. Figure 18 shows the acceleration (deceleration) and speed of the autonomous vehicle. Approximately at the 6.1 s the AV decelerates at –4m/s^2 because the collision is occurring in the IC, then the CC = IC, at the 8.13 s the AV decelerates at normal conditions (–2 m/s^2) because the collision is occurring a the current intersecting coordinate, at the 10.17 s the AV accelerates to continue with its path.

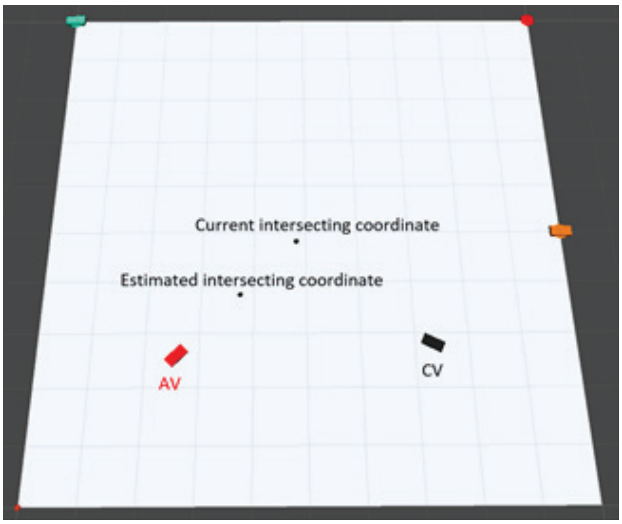

Figure 16. Simulation with an AV and the CV

http://dx.doi.org/10.22201/fi.25940732e.2019.20n2.020

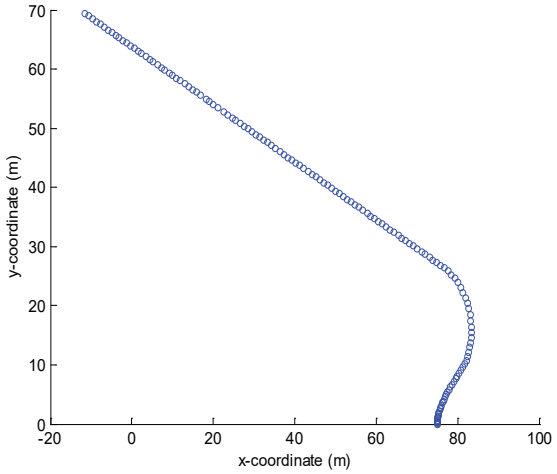

Figure 17. Controlled vehicle trajectory

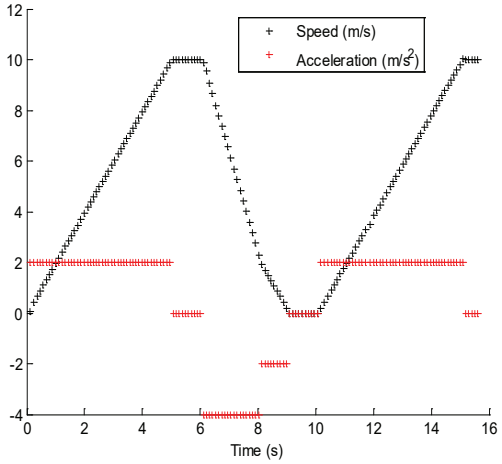

Figure 18. Autonomous vehicle variables

Conclusions

Adaptive solutions for two problems were tested: 1) setting the traffic lights times and 2) choosing optimal routes, both suitable to manage autonomous traffic, however combining these (out of the scope in this study) a major benefit is expected.

With the procedure presented in the intersection control section, the traffic light times of intersecting streets are effectively coordinated in the benefit of a better traffic flow. In the simulations conducted in a scenario with two intersections, in a lapse time of 600 s, the vehicles passing the last intersection from the south-north direction (which has the maximum arrival vehicles frequency, Ts=1) with the adaptive control were 462, with the conventional control were 412, with a difference of 50 vehicles in favor of the adaptive. In the simulations performed in a scenario with one intersection and a simulation time of 300 s, for the vehicles circulating from south to north (Ts=0.5 s), whith the adaptive control 289 vehicles passes the intersection, the conventional control registers 201 vehicles, with 88 vehicles in favor of the adaptive. For the vehicles circulating from north to south (Ts= 1 s) the difference is 20 vehicles in favor of the adaptive.

The routing algorithm, designed to select the streets that conform a viable path to the destination, uses the streets density with the aim to improve the vehicles travel time (because it is faster to travel on less congested streets). The algorithm first seeks the streets (that conform a path) with the lowest initial density selected, this threshold is increased until a path is conformed of streets with lower density (or equal) than the threshold, guaranteeing an optimal

route. In the simulation test it was validated that the algorithm guides a vehicle through the lest congested streets, which result in a shorter travel time of the guided vehicle vs. not guided.

The adaptive methods presented in this study, 1) for setting the traffic lights times and 2) selecting the streets to reach a destination, are flexible in the sense that are able to modify their control actions in benefit of the current traffic. In addition, our approaches not depend on a vehicle to vehicle communication, instead the control sets the traffic lights (first case) or communicates a suitable route for traveling (second case).

The results from the autonomous vehicles interactions section suggest that partially automate the traffic, in a controlled environment, is possible. In the simulation 1, the braking algorithm regulates the AVs acceleration (deceleration) to avoid collisions if trajectories with a common coordinate are detected. If the CV participates, the future position of the CV is estimated, and which vehicle has the right of way is assigned. In the simulation 2 it was proved that the rules to control the driving interaction among the AVs work together with the rules to set the priority between the controlled and autonomous vehicles. In the simulation 3 the CV circulates without restrictions, this vehicle changes its traveling direction and the AV reacts and decelerates opportunely, since the collision coordinate is re-calculated depending the previous position coordinates of the CV. The time cycle to refresh data is 1 s, an improvement over (Qian *et al*., 2014), where 0.05 s is required.
To extend the findings of this study to real traffic, it will be required to modify the design considering the limitations of the selected sensors, the delay in communications, and the noise (unexpected occurrences). The algorithm to regulate the driving interaction between autonomous vehicles (and the manually controlled) was tested in simulations with a limited number of vehicles. The future work is to perform simulations with more vehicles (autonomous and non).

Annex A

Routing algorithm MATLAB code.

```
clear all; clc; double inc;
con=0;vf=0;n=0;nt=0;u1=1; u2=1; l1=1;l2=1;k=1;c=0;costo
=0;costoT=200*12; n=4;inc=0;
A = zeros(12,1); B = zeros(12,1); l=zeros(3,2);
u=zeros(3,2);lt=zeros(3,2); ut=zeros(3,2);

temp=dec2bin((0:2^n-1));
for i=1:2^n
   for j=1:n
      t(i,j)=str2num(temp(i,j));
   end
end
i2=1;
for i=1:2^n
   if (sum(t(i,:))== 2)
      t2(i2,:) = t(i,:);
      i2=i2+1;
   end
end
t2;c1=size(t2);c2=c1(1,1);

c=0;con=0;
while (vf==0)

for j=1:12
if A(j,1)<=con
   B(j,1)=1;
else
   B(j,1)=0;
end
end
for i2=1:c2
 k=1;t=t2(i2,:);
l=zeros(n-1,n-2);
u=zeros(n-1,n-2);

    if t(1,1)==0
       l(l1,l2)=1 ;
    end
    if t(1,1)==1
       u(u1,u2)=1;
    end

 for i=2:n

   if t(1,i)==0
   l2t=sum(t(1:k)==0);
   l1t=sum(t(1:k)==1);
   l(1+l1t,1+l2t)=1;
   end

   if t(1,i)==1
      u1t=sum(t(1:k)==0);
      u2t=sum(t(1:k)==1);
      u(1+u1t,1+u2t)=1 ;
   end
k=k+1;
 end
 if(B(1,1)==1&&u(1,1)==1)c=c+1;costo=costo+A(1,1);end
 if(B(2,1)==1&&u(1,2)==1)c=c+1;costo=costo+A(2,1);end
 if(B(3,1)==1&&l(1,1)==1)c=c+1;costo=costo+A(3,1);end
```

```
 if(B(4,1)==1&&l(2,1)==1)c=c+1;costo=costo+A(4,1);end
 if(B(5,1)==1&&l(3,1)==1)c=c+1;costo=costo+A(5,1);end
if(B(6,1)==1&&u(2,1)==1)c=c+1;costo=costo+A(6,1);end
if(B(7,1)==1&&u(2,2)==1)c=c+1;costo=costo+A(7,1);end
 if(B(8,1)==1&&l(1,2)==1)c=c+1;costo=costo+A(8,1);end
 if(B(9,1)==1&&l(2,2)==1)c=c+1;costo=costo+A(9,1);end
                    if    (B(10,1)==1    &&    l(3,2)==1)
c=c+1;costo=costo+A(10,1);end
                    if    (B(11,1)==1    &&    u(3,1)==1)
c=c+1;costo=costo+A(11,1);end
                    if    (B(12,1)==1    &&    u(3,2)==1)
c=c+1;costo=costo+A(12,1);end

  if c>=4
   if (costo<costoT)
     costoT=costo; ut=u; lt=l;
   end
  end
c=0;costo=0;
end

con=con+1;
if (con>200)
   vf=1;break;
end
end
```

Annex B

The braking algorithm was written in C# and is presented in three parts: 1) disabling AVs rules, 2) enabling AVs rules and, 3) the user-controlled vehicle rules. The position 0 of an array is referring a variable related with car_1, position 1 for car_2 and so on. The terms used in the algorithm are presented in Table B1.

Table B1. Braking algorithm notation

$adis_{ij}$ is dis_{ij} measured at the previous step time.
adiscl is *dislc* at the previous step time.
active flag used to enable or disable the priority of a vehicle.
active2 flag used to determine if an AV was disabled to move.
accelerationc the acceleration of the autonomous vehicle.

adisij is disij measured at the previous step time.
adiscl is dislc at the previous step time.
active flag used to enable or disable the priority of a vehicle.
active2 flag used to determine if an AV was disabled to move.
accelerationc the acceleration of the autonomous vehicle.
accelerationl the acceleration of the controlled vehicle.
calculation() function that takes three arguments (the distance to the intersecting coordinate, the current speed and acceleration), it returns the time required by the vehicle (the AV or the CV) to arrive at the intersection coordinate.
df= v / -(a*2), is the distance required to brake the vehicle to a full stop, a=-2 is the deceleration.
disij the distance between the i vehicle and the intersecting coordinate whit the j vehicle.
discl the distance between the AV and the cross point with the CV.
dislc the distance between the CV and the cross point with the AV.

drive flag used to enable or disable the vehicle motion, if is equal to true the vehicle accelerates, otherwise decelerates.
disable2 flag used to control the activation of the decelerated vehicles.
disable3 used to accumulate the number of flags required in true to enable the AV motion.
i, j, z, k counters to refer the index of the variables involved of an AV.
ms is the maximum speed.
num is the number of autonomous vehicles in the simulation.
num2 = (num* (num - 1)) / 2, is the number of possible combinations in pairs between the autonomous vehicles, (for 3 vehicles, there are three combinations: [0,1], [0,2] and [1,2])
pass() function used to check if the AV has passed or not the cross point with the CV.
sd=8 m is a safety distance to stop with anticipation.
speedc the speed of the referred autonomous vehicle.
speedl the speed of the controlled vehicle.
t is the step time.
timec the time required by the AV to arrive at the cross point with the CV.
timel ithe time required by the CV to arrive at the intersection with the AV.
v is the current speed.
va the speed at the previous step time.
x is the current traveled distance.
xa the distance traveled measured at the previous step time.

Braking algorithm (part 1), disabling autonomous vehicles.

Initial conditions:

```
for (int k=0; k<num2*2; k++)  {active[k]=true; disable2[k]=true}
for (int k=0; k<num; k++)  {active2[k]=true;}
```

```
#1        z=0;
#2        for (i=0; i<num; i++) {
#3        for (j=i+1; j<num; j++) {
#4          if (dis [z] - sd <= df [i] && active [z] == true ) {
#5          active [z + 1] = false;
#6            if (dis [z + 1] - sd <= df [j]) {
#7              drive [j] = false; disable2[z]=false;
#8            } //end of
#9          } //end of
#10         if (dis [z + 1] sd <= df [j] && active [z + 1] == true ) {
#11            active [z] = false;
#12           if (dis [z] - sd <= df [i]) {
#13             drive [i] = false; disable2[z+1]=false;
#14           } // end of
#15         }  // end of
#16         z=z+2;
#17       } //end for j
#18       }  //end for i
```

Braking algorithm (part 2), enabling autonomous vehicles.

```
#1        z = 0; for (int k=0; k<num; k++) {disable3[k]=0;}
#2        for (int i=0; i<num; i++) {
#3        for (int j=i+1; j<num; j++) {
```

```
#4
#5        if (((adis [z] - dis [z]) < 0) && dis[z]>5 && active [z + 1] == false)
#6        { disable2[z]=true;}
#7
#8        if (((adis [z+1] - dis [z+1]) < 0) && dis[z+1]>5 && active [z] == false)
#9        { disable2[z+1]=true;}

#10       for (k=0;k<num;k++)
#11       {
#12         if (i==k && disable2[z+1]==true)
#13         {disable3[k]= disable3[k]+1;}
#14            if (j==k && disable2[z]==true)
#15           {disable3[k]= disable3[k]+1;}
#16
#17             if (disable3[k]==(num-1))
#18               {drive[k] = true;}
#19       }//end for
#20       z=z+2;
#21       }//end for j
#22       }//end for i
```

Braking algorithm (part 3), the user-controlled vehicle rules.

```
#1        for (k=0; k<num; k++) {
#2          if ((dislc [k] - adislc [k] <= 0) && (discl [k] - adiscl [k] <= 0)) {

#3            if (speedc[k] > .01) {
#4            calculation (discl [k] + 5, speedc[k],accelerationc[k]);
#5            timec [k] = time; }//end if
#6      else{ timec [k] = 0;}

#7            if (speedl > .01) {
#8            calculation (dislc [k]-5, speedl, accelerationl);
#9            timel [k] = time; }//end if
#10             else {timel [k] = 0;}

#11       if ((((discl[k]-sd≤df[k]) && dislc[k]<5) || ((discl[k]-sd≤df[k])
#12          && timel[k]<timec[k] && timec[k]>0 && timel[k]>0)))
#13  {drive[k]=false;active2[k]=false}

#14  if (active2 [k] == false) {drive[k]=false;}

#15 if (((dislc[k]-adislc[k]≥0) && (dislc[k]>5) && active2[k]==false)
          ||   (discl [k]-(sd*2)>df [k] && active2[k]==false))
#16  {drive[k]=true;active2[k]=true;}
#17 }//end for k
```

Calculation function.

```
void calculation (p1,p2,p3)
{va=p2;a=p3;t=0.1;xa=0;time=0;v=0;
while (x <= p1) {
if (v >= ms) { a = 0; va = ms;}
v = (va) + (a * t);
x = xa + va * t + 0.5 * a * t * t;
va = v; xa = x; time = time + t;
}//end while
}//end calculation
```

References

Biham, O., Middleton, A.A., Levine, D. (1992). Self-organization and a dynamical transition in traffic-flow models. *Physical Review A*., 46(10), R6124. Recovered from https://doi.org/10.1103/PhysRevA.46.R6124

Carrillo-González, J.G., Arámburo-Lizárraga, J., Barbosa-Santillán, L.I. (2018). Acceleration (Deceleration) Model Supporting Time Delays to Refresh Data. *Promet-Traffic&Transportation*, 30(2), 141-149.

De Gier, J., Garoni, T.M., Rojas, O. (2011). Traffic flow on realistic road networks with adaptive traffic lights. *Journal of Statistical Mechanics: Theory and Experiment*, (04), P04008.

Fouladvand, M.E., Sadjadi, Z., Shaebani, M.R. (2004). Optimized traffic flow at a single intersection: traffic responsive signalization. *Journal of Physics A: Mathematical and General*, 37(3), 561.

Gershenson, C. (2004). Self-organizing traffic lights. ArXiv Preprint Nlin/0411066. Recovered from https://arxiv.org/ct?url=https%3A%2F%2Fdx.doi.org%2F10.1007%252F978-1-84628-982-8_3&v=d158f5fa

Gershenson, C. & Rosenblueth, D.A. (2012a). Adaptive self-organization vs static optimization: A qualitative comparison in traffic light coordination. *Kybernetes*, 41(3/4), 386-403. Recovered from https://doi.org/10.1108/03684921211229479

Gershenson, C. & Rosenblueth, D.A. (2012b). Self-organizing traffic lights at multiple-street intersections. *Complexity*, 17(4), 23-39. Recovered from https://doi.org/10.1002/cplx.20392

Helbing, D., Lämmer, S., Lebacque, J.-P. (2005). Self-organized control of irregular or perturbed network traffic. In Optimal control and dynamic games (pp. 239-274). Springer.

Lämmer, S. & Helbing, D. (2008). Self-control of traffic lights and vehicle flows in urban road networks. *Journal of Statistical Mechanics: Theory and Experiment*, (04), P04019. Recovered from https://doi.org/10.1088/1742-5468/2008/04/P04019

Liao, T. (2012). *Uav collision avoidance using a* algorithm* (master's thesis). Auburn University.

Nagel, J., Trepagnier, P.G., Koutsougeras, C., Kinney, P.M., Dooner, M. (2006). The Culebra algorithm for path planning and obstacle avoidance in Kat-5. In null 247-253. IEEE.

Qian, X., Gregoire, J., Moutarde, F., De La Fortelle, A. (2014). Priority-based coordination of autonomous and legacy vehicles at intersection. In Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on 1166-1171.

Resende, P. & Nashashibi, F. (2010). Real-time dynamic trajectory planning for highly automated driving in highways. In Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on 653-658. Recovered from https://doi.org/10.1109/ITSC.2010.5625194

Technologies, U. (n.d.). unity3d. Recovered from https://unity3d.com/

Tonguz, O.K. & others. (2011). Biologically inspired solutions to fundamental transportation problems. *IEEE Communications Magazine*, 49(11), 106-115.

Viriyasitavat, W. & Tonguz, O.K. (2012). Priority Management of Emergency Vehicles at Intersections Using Self-Organized Traffic Control. In VTC Fall 1-4. Recovered from https://doi.org/10.1109/VTCFall.2012.6399201

Wei, J. & Dolan, J.M. (2009). A robust autonomous freeway driving algorithm. In Intelligent Vehicles Symposium, 2009 IEEE 1015-1020. Recovered from https://doi.org/10.1109/IVS.2009.5164420

Wischoff, L., Ebner, A., Rohling, H., Lott, M., Halfmann, R. (2003). SOTIS-a self-organizing traffic information system. In Vehicular Technology Conference, VTC 2003-Spring. The 57th IEEE Semiannual (4), 2442-2446. Recovered from https://doi.org/10.1109/VETECS.2003.1208829

Yu-qin, F., Jun-qiang, L., Zhong-Yu, X., Gui-e, Z., & Yi, H. (2013). Route choice model considering generalized travel cost based on game theory. *Mathematical Problems in Engineering*. Recovered from http://dx.doi.org/10.1155/2013/464038