Virtual Belt Algorithm for the Management of Isolated Autonomous Intersection

Chentong Bian, Guodong Yin *, Liwei Xu and Ning Zhang

School of Mechanical Engineering, Southeast University, Nanjing 211189, China; 230149014@seu.edu.cn (C.B.); 230149307@seu.edu.cn (L.X.); nzhang_cn@seu.edu.cn (N.Z.)
* Correspondence: ygd@seu.edu.cn; Tel.: +86-025-5209-0501 (ext. 8333)

Received: 8 October 2018; Accepted: 2 November 2018; Published: 8 November 2018

Abstract: To enhance traffic efficiency, in this paper, a novel virtual belt algorithm is proposed for the management of an isolated autonomous intersection. The proposed virtual belt algorithm consists of an offline algorithm and an online algorithm. Using the offline algorithm, the considered intersection can be modeled as several virtual belts. The online algorithm is designed for the real-time application of the virtual belt algorithm. Compared with the related algorithms, the main advantage of the proposed algorithm is that, there are several candidate trajectories for each approaching vehicle. Thus, there are more opportunities for an approaching vehicle to obtain a permission to pass an intersection, which is effective to improve traffic efficiency. The proposed algorithm is validated using numerical simulations conducted by Matlab and VISSIM. The simulation results show that the proposed algorithm is effective for autonomous intersection management.

Keywords: autonomous intersection; intersection management; virtual belt algorithm; isolated intersection; autonomous vehicle

1. Introduction

Intersection is one of the key elements of a modern traffic system. A great deal of time was wasted to wait for green lights, which brought huge economic losses [1]. Traffic light is a conventional method to control traffic flows at intersections. Generally, the traffic lights with several signal phases are applied to coordinate conflicting traffic flows. While due to the limitations of traffic lights (i.e., the minimum and maximum green times), an approaching vehicle may have to spend some extra time for green lights even there are no potential collisions. These limitations not only decrease traffic efficiency but also cost unnecessary time of passengers.

In recent years, the Dedicated Short Range Communication (DSRC) technology has greatly facilitated the vehicle-to-vehicle and vehicle-to-infrastructure communication [2–7]. Based on DSRC, autonomous vehicles can be connected and share their information with other vehicles and infrastructures. Thus, CAVs can be coordinated simultaneously to improve traffic efficiency [8]. Autonomous intersection, which is designed based on DSRC and CAVs, has drawn much attention around the world. A variety of algorithms for the management of an isolated autonomous intersection have been proposed to improve traffic efficiency without traffic lights.

According to the method used to coordinate approaching CAVs, the previous algorithms can be classified as global-trajectory-optimization based algorithms and non-global-trajectory-optimization based algorithms. For the global-trajectory-optimization based algorithms, the decision variables of the trajectories of many CAVs (at least including all CAVs in collision zones) are used to construct an optimization problem. These trajectories are optimized simultaneously using iteration calculation to determine the trajectories of these CAVs. For the no-global-trajectory-optimization based algorithms,
there is no such an optimization problem. The permissions of CAVs to pass an intersection are determined by a predefined policy.

The global-trajectory-optimization based algorithms mainly include model predictive control [9], genetic algorithm [10], mix-integer non-linear programming [11], mix-integer linear programming [12,13], integer linear programming [14], linear programming [8], etc. Using optimization, these algorithms can plan the trajectories of the CAVs at an intersection simultaneously to improve traffic efficiency. The main advantage of these algorithms is that the optimal calculation can be used to find an optimal or a local optimal solution for intersection management. Thus, theoretically, these algorithms show better performance than the non-global-trajectory-optimization-based algorithms. The major disadvantage of these algorithms is the high requirement for calculation. Due to the coupling of vehicle trajectories, these algorithms have to simultaneously control all CAVs at an intersection (at least within a certain range including all conflict zones), which leads to a high requirement for calculation from two aspects. First, the count of decision variables of optimal calculation increases with the count of CAVs, which aggravates the calculation burden. Second, even the trajectories of all CAVs at collision zones have been planned, these trajectories have to be planned again when a new approaching CAV is found. Thus, the trajectory of a CAV may be planned again and again, which further increases the high requirement for calculation.

The non-global-trajectory-optimization based algorithms mainly include cooperative adaptive cruise control [15,16], multi-agent [1,17], priorities and brake-safe control based algorithm [18], auction [19] (or market [20]) based algorithm, buffer-assignment mechanism algorithm [21], platoon based algorithm [22], slots preassigning [23], virtual traffic light [24], and the combination of these algorithms [25], etc. These algorithms can improve traffic efficiency by coordinating the orders of CAVs to pass intersections. The main advantage of these algorithms is the low computation burden. Thus, they can be easily applied for the intersections with a large number of CAVs. The disadvantage is that there is only one candidate trajectory for a CAV during a control circle. If there are potential collisions for a CAV, the CAV will have to slow down or stop to wait for a possible permission next control circle. Thus, there are still some potentials to improve traffic efficiency if more trajectories are designed and compared in a control circle.

Aiming to enhance traffic efficiency, a VB algorithm is proposed for the management of an isolated autonomous intersection. The proposed algorithm is divided into an offline algorithm and an online algorithm. By the proposed algorithm, most of the calculation caused by the coupling of vehicle trajectories can be done offline. There is no optimization problem to calculate the trajectories of all CAVs simultaneously in the VB algorithm. Thus, the VB algorithm is a no-global-trajectory-optimization based algorithm. Compared with previous no-global-trajectory-optimization based algorithms, the main advantage of the proposed algorithm is that, there are several candidate trajectories can be chose for a CAV to pass an intersection in a control circle. Thus, there are more opportunities for an approaching CAV to get a permission to pass an intersection. By contrast, for previous no-global-trajectory-optimization based algorithms, in a control circle, there is only one candidate trajectory can be used for an approaching CAV.

As shown in Figure 1, the proposed VB algorithm consists of an online algorithm and an offline algorithm. As indicated by the name, the offline algorithm is used before the online application of the algorithm. The offline algorithm is comprised of VBs modeling and collision checking. By the method of VBs modeling, the considered intersection is modeled as several VBs. There are several VGs that fixed on each VB. The collision relations among these VGs can be obtained by collision checking. The online algorithm is used for the online application of the VB algorithm, which is designed based on the principle of “first-in-first-service” [1]. The online algorithm consists of VG selecting and trajectory planning. Several suitable VGs can be obtained by the method of VG selecting. By the trajectory planning method, the most suitable VG and the collision free trajectory can be found for each approaching CAV. Following the planned trajectory, an approaching CAV can pass an intersection without collisions.
The remainders of the paper are organized as follows: In Section 2, the problem of the considered intersection management is illustrated. In Section 3, several basic conceptions of the VB algorithm are introduced, which lay the foundation of this algorithm. In Section 4, the method to obtain the collision relation among the VGs in different VBs is proposed. In Section 5, the initial ideal of VB algorithm is discussed, which provides a simple demonstration of the VB algorithm. In Section 6, the VB algorithm designed for actual application is presented. In Section 7, the numerical simulations of the VB algorithm are conducted to demonstrate the effectiveness of this algorithm. At last conclusion is given in Section 8.

2. Problem Formulation

In this study, an inhomogeneous four-road intersection is considered as shown in Figure 2. The roads in the intersection are labeled with “E”, “N”, “W” and “S”, respectively. The paths of vehicles at the intersection are plotted in Figure 3. This figure shows that there are many cross points among these paths. The proposed VB algorithm can be easily extended to a homogeneous intersection. The inhomogeneous intersection considered in this study is used to represent a more general case.
2.1. Components of Intersection Management System

The main components of the intersection management system include an IMU and CAVs. The communication between the IMU and a CAV can be achieved by vehicle-to-infrastructure communication. The IMU is responsible for the trajectory planning of the CAVs at intersection. A CAV should send a request for passing the intersection and follow the trajectory planned by the IMU. The request of a CAV is called a RCAV for short. Several assumptions are made to facilitate the discussion of intersection management. These assumptions are feasible in technology. Some of these assumptions are commonly made in previous studies [1,8,14].

- The position, velocity and dynamic limits of a CAV can be transmitted to the IMU.
- A trajectory planned by the IMU can be sent to the corresponding CAV.
- A CAV can follow the trajectory planned by the IMU within the limits of the CAV.
- The mean speed of the considered traffic flow is much lower than the maximum speed of the vehicle with low dynamic performance. Generally, the maximum speed of vehicles on road is higher than 90 km/h. However, the mean speed of the traffic flow on city environment is much lower than 60 km/h.
- Before the intersection, each approaching vehicle has turned to the corresponding lane according to its destination.

2.2. Working Process of Intersection Management Unit

To determine the passing orders of CAVs, the CAVs at intersection are divided into a waiting group and a passing group. The CAVs in the waiting group are the CAVs whose trajectories of passing the intersection have not been planned, which are called waiting CAVs. The CAVs in the passing group are the CAVs whose trajectories have been planned successfully to pass the intersection, which are called passing CAVs. The RCAVs of the CAVs in the waiting group are organized into a queue based on the principle of first-in-first-service.

The working process of the IMU is shown in Figure 4. First, the IMU finds the first waiting CAV in the waiting group. Then the trajectory of the CAV is planned by the IMU using the proposed online algorithm. At last the planned trajectory is sent to the corresponding CAV. The IMU repeats the same
working process to coordinate approaching CAVs to pass the intersection. The waiting CAVs should slow down before the finding of the planned trajectory.

![Diagram of IMU](image)

**Figure 4. Working process of the IMU.**

3. Basic Concepts for Virtual Belt Algorithm

In this section, several concepts for the VB algorithm are introduced. These concepts include virtual path, virtual grid, virtual belt, loop add and loop subtract, which are the foundation of the proposed VB algorithm.

3.1. Virtual Path and Virtual Grid

As shown in Figure 5, there are a VP and a VG in this figure. VP is a directed continuous curve with a start point \( (p_s) \) and an end point \( (p_e) \). The length of VP is denoted by \( l_p \). VG is a grid which consists of four edges in geometry space. A VG moves circularly from \( p_s \) to \( p_e \) along a VP with a constant speed of \( v_g \). \( l_g \) and \( l_w \) mean the length and width of the VG, respectively. The nearest edge to \( p_s \) is defined as the start edge of the VG. The start edge is represented by \( e_s \) as shown in Figure 5.

![Diagram of VP and VG](image)

**Figure 5. Virtual path and virtual grid.**

The motion of a VG is plotted in Figure 6. At the time of \( t_0 \), the center of \( e_s \) coincides with \( p_s \). The VG is called to be born at the time of \( t_0 \). The center of \( e_s \) will coincide with \( p_e \) at the time of \( t_d \). Where \( t_d = t_0 + t_c \), \( t_c = \frac{l_p}{v_g} \) is defined as the time circle of the VG, \( t_g = \frac{l_g}{v_g} \) is defined as the grid time of the VG. At the time of \( t_d \), the center of \( e_s \) coincides with \( p_e \) and the VG disappears, which is called the death of the VG. The VG will appear and move forward to \( p_e \) at the time of \( t_d \) as well. Thus, the VG moves circularly from \( p_s \) to \( p_e \) with the time circle of \( t_c \). Consider a set of VGs. If these VGs are of the same speed and size, and the lengths of the VPs of these VGs are equal as well, these VGs are called uniform. A VG can be viewed as a virtual container that appears at \( p_s \), and moves to \( p_e \) with the speed of \( v_g \). When a VG arrives at \( p_e \), the VG returns to \( p_s \) in no time. The VG repeats the same motion along the time.
3.2. Virtual Belt

VB is a belt that consists of several close-spaced VGs along the same VP. There are no clearances or overlaps of these VGs. The total count of the VGs of a VB is denoted as \( C_g \). All VGs in a VB are uniform. These VGs are represented by \( G_{vi} \). A VB with 7 VGs which are labeled from 1 to 7 is shown in Figure 7. The motion of the VB is demonstrated in Figure 7 as well. When a VG in the VB is dead at \( p_e \), the VG will be born at \( p_s \). In this figure, VG 7 is dead and born at \( t_0 + t_g \). VG 6 is dead and born at \( t_0 + 2 \cdot t_g \). Note that as the lengths and speeds of these VGs are equal, \( G_{vi} \) will be the same place of \( G_{vi} \) after the time of \( t_g \). Similar to VG, VB is defined in geometry space instead of in actual world.

![Figure 6. Motion of virtual grid.](image)

3.3. Loop Add and Loop Subtract

The mathematic functions loop add (represented by \( +_L \)) and loop subtract (represented by \( -_L \)) are defined for the calculation of integrators. Based on these functions, the collisions among VGs can be discussed in a simple way, which will be demonstrated in the following section. Consider two integrators \( i \) and \( j \) (\( 1 \leq i, j \leq L \), \( L \) is an integrator and \( 2 \leq L \)). The function of loop add is defined as
For example, one has 

\[ 3 + 3 = 3 + 2 = 2. \]

Similarly, the function of loop subtract is defined as

\[
i - L = \begin{cases} 
  i - j, & (i - j) \geq 1 \\
  i - j + L, & (i - j) < 1
\end{cases}
\]

For instance, one has 

\[ 1 - 3 = 1 - 2 + 3 = 2. \]

### 4. Collision Checking of Virtual Belts

In this section, the method to find the collision relation among the VGs of different VBs is shown. In certain condition, a very interesting phenomenon is that some VGs never collide with each other and some always collide with each other in each time circle.

#### 4.1. Collision Checking between Virtual Grids

Consider two uniform VGs (\(G_v\) and \(G'_v\)). \(G_v\) is born at \(t_0\) and \(G'_v\) is born at \(t_0 + \Delta t\). Depending on the VPs and the born times of these VGs. During the time of \((t_0 + t_{\Delta t}, t_0 + t_{c} + t_{\Delta t})\), there are only two possible results of collision checking between the two VGs. The first one is that \(G_v\) collides with \(G'_v\) in a certain time as plotted in Figure 8. The second one is that \(G_v\) never collides with \(G'_v\) as shown in Figure 9.

![Figure 8](image-url)

**Figure 8.** Motion of two VGs with collision, (a) \(t = t_0 + \Delta t\), (b) \(t = t_0 + \frac{1}{2}t_c + \Delta t\), (c) \(t = t_0 + \frac{3}{2}t_c + \Delta t\), (d) \(t = t_0 + \frac{3}{2}t_c + \Delta t\), (e) \(t = t_0 + \frac{7}{2}t_c + \Delta t\), (f) \(t = t_0 + t_c + \Delta t\).

These VGs move along the corresponding VPs with the same speed. If \(G_v\) does not collide with \(G'_v\) during the time range of \((t_0 + t_{\Delta t}, t_0 + t_{c} + t_{\Delta t})\), then \(G_v\) will not collided with \(G'_v\) at the time range of \((t_0 + t_{c} + t_{\Delta t}, t_0 + 2t_{c} + t_{\Delta t})\). Consequently, it can be deduced that \(G_v\) and \(G'_v\) will not collide forever. Similarly, if \(G_v\) collides with \(G'_v\) during the time range of \((t_0 + t_{\Delta t}, t_0 + t_{c} + t_{\Delta t})\), then \(G_v\) will collide with \(G'_v\) at each time circle in future. In another word, the future collision between two uniform VGs can be predicted based the result of collision checking in a time circle.
The collision checking algorithm for two uniform VGs is shown in Algorithm 1.

**Algorithm 1** Collision checking of two VGs.

**Input:** two VGs ($G_v$ and $G'_v$)

**Output:** whether two VGs will collide with each other in the future

1. Initial the positions of $G_v$ and $G'_v$
2. Divide the time circle ($t_{\text{init}}$, $t_{\text{init}} + t_c$) into small enough segments: $t = (t_{\text{init}}, t_{\text{init}} + t_c, t_{\text{init}} + 2t_c, \cdots, t_{\text{init}} + t_c)$
3. for $t$ in $t$
4. Calculate the positions of $G_v$ and $G'_v$ at the time of $t$;
5. if $G_v$ and $G'_v$ collide with each other then
6. return true
7. end if
8. end for
9. return false

If $G'_v$ collides with $G_v$ in a time circle, $G'_v$ is called a CVG of $G_v$. The total CVGs of a VG is defined as the CVGR of the VG. CVGR provides a simple but effective method to record the collisions among VGs. As all VGs repeat the same motion, a CVG of $G_v$ collides with $G_v$ in each time circle. And a VG that not belongs to the CVGs of $G_v$ will not collide with $G_v$ forever.

4.2. Collision Checking of Virtual Grid and Virtual Belt

Given a set of VGs (denoted as $S$, the VGs in $S$ are represented by $G_{v1}, G_{v2}, \cdots$), the CVGR of a VG (denoted as $G_{v}$) in these VGs can be obtained through Algorithm 2.

Consider two VGs (which are labeled as $G_{v1}$ and $G_{v2}$, respectively) and a VB (labeled as $B_v$) as shown in Figure 10. The VPs of these VGs and VB are plotted with solid red line in that figure. The coordinates of the start and end points of these VPs are shown in that figure as well. The unit of the coordinate is meter. The lengths of the three VPs are 72 m. There are twelve VGs on $B_v$, which are represented by $G^B_{vi,i=1,2,\cdots,12}$. All of these VGs are uniform. $l_g = 6$ m, $l_w = 4$ m, $v_g = 2$ m/s. $t_c = \frac{t_g}{v_g} = 36$ s.
**Algorithm 2 Calculation of the CVGR of A VG.**

**Input:** a set of VGs (S) and a VG (G_v)

**Output:** the CVGR of G_v

1. Initial an empty set S_{G_v}
2. for G_{vi} in S do
3.   call Algorithm 1. Collision checking of G_{vi} and G_v
4.   if G_{vi} collides with G_v then
5.     append G_{vi} to S_{G_v}
6.   end if
7. end for
8. return S_{G_v}

---

**Figure 10.** Collision checking of a VB and two VGs.

Several camera shots of B_v, G_v1 and G_v2 between 28 s and 33 s are plotted in Figure 11. Using Algorithm 2, it can be found that G_v1 collides with G_{B_v1}, G_{B_v12} and G_{B_v11}. G_v2 collides with G_{B_v2}, G_{B_v1} and G_{B_v12}. The CVGR of G_v1 can be organized as \{G_{B_v1}, G_{B_v12}, G_{B_v11}\}. The CVGR of G_v2 is \{G_{B_v2}, G_{B_v1}, G_{B_v12}\}. Similarly, one can obtain that the CVGR of G_{B_v3} is empty.
Figure 11. Motion of a VB and two VGs Motion of a VB and two VGs, (a) t = 28 s, (b) t = 29 s, (c) t = 0 s, (d) t = 31 s, (e) t = 32 s, (f) t = 33 s.

4.3. Collisions Checking of Virtual Belts

If the VGs in several VBs are uniform, the CVGR of each VG on these VBs can be obtained by Algorithm 2.

Consider three VBs as shown in Figure 12. The coordinate unit is meter. These VBs are denoted by $B_{v_{j}}$, $j = 1, 2, 3$. The VGs of $B_{v_{j}}$ are represented by $G_{v_{i}}^{B_{j}}$, respectively. $C_g = 12$ means the total count of the VGs in a VB. The VGs in these VBs are uniform. As demonstrated previous, the CVGR of $G_{v_{1}}^{B_{2}}$ is $\{G_{v_{1}}^{B_{1}}, G_{v_{1z}}^{B_{1}}, G_{v_{11}}^{B_{1}}\}$. Note that $G_{v_{i}}^{B_{j}}$ will be the same position of $G_{v_{i}}^{B_{j}}$ after the time of $t_g$. If $G_{v_{i}}^{B_{j}}$ collides with $G_{v_{x}}^{B_{y}}$ which is a CVG of $G_{v_{i}}^{B_{j}}$ at the time of $t$, then $G_{v_{(i+C_g)}}^{B_{y}}$ will collide with $G_{v_{(i+C_g)}}^{B_{x}}$ at the time of $t + t_g$. Thus, the CVGR of $G_{v_{1z}}^{B_{2}}$, note that $G_{v_{i}}^{B_{j}} = G_{v_{(i+C_g)}}^{B_{j}}$, is $\{G_{v_{12}}, G_{v_{11}}, G_{v_{10}}\}$. The CVGR of $G_{v_{i}}^{B_{2}}$ can be summarized as $\{G_{v_{i}}^{B_{1}}, G_{v_{(i-C_g)}}^{B_{1}}, G_{v_{(i-C_g)}}^{B_{1}}\}$ and $\{G_{v_{i}}^{B_{2}}, G_{v_{(i+C_g)}}^{B_{2}}, G_{v_{(i+C_g)}}^{B_{2}}\}$, respectively.
5. Initial Idea of Virtual Belt Algorithm

In this section, the initial ideal of the proposed algorithm is discussed. Then an example to demonstrate the idea for intersection management is shown. At last several deficiencies of the initial ideal are discussed. This example demonstrates the basic principle of the proposed VB algorithm, which is made to facilitate the understanding of the core of the proposed VB algorithm.

5.1. Discussion of The Initial Idea

Consider that $G_v$ and $G'_v$ (as shown in Figure 9) are placed on an intersection with two through lanes. The start and end points of these VPs can be used to represent the departures and destinations of several approaching CAVs. The size of the VG is bigger than these CAVs. It is required that these CAVs should follow the trajectories of $G_v$ and $G'_v$ to their destinations one by one without collision.

If a CAV is required to follow the trajectory of a VG to through the intersection, the CAV is called to be transferred by the VG and the VG is called assigned with the CAV as well. When a CAV arrives at the destination, it should move away from the intersection in constant speed to avoid affecting the “transfer service” of the intersection.

The VGs used for the transfer of CAVs should be selected carefully. An inappropriate choice may lead to collisions among CAVs. For instance, if the VGs demonstrated in Figure 8 are used for CAV transfer, the approaching CAVs may collide with each other. If a VG and its CVGs are used for the transfer of CAVs simultaneously, collisions among these CAVs may occur. A natural idea is that when a VG is used to transfer a CAV, its CVGs should not be used. Thus, the collisions among CAVs can be avoided at the considered intersection.

The starting point of the initial idea is that several VBs are placed on the considered intersection. Each approaching CAV is assigned with a VG before the conflict points in the intersection. Thus, the approaching CAVs can follow the trajectories of the selected VGs and pass the intersection without collisions. Two requirements should be satisfied to ensure the safety of intersection management: First, the VGs which are assigned with CAVs should not collide with each other at the intersection. Second, the CAVs should be able to follow the planned trajectories with small tracking error.

Figure 12. Collision checking of three VBs.
5.2. Demonstration of the Initial Idea

A simplified intersection with three through lanes is shown in Figure 13. There are two approaching CAVs for each lane. According to the time of getting entry to the intersection, the orders of these CAVs are labeled from 1 to 6 as shown in this figure. The length and width of each lane are 120 m and 4 m, respectively. The length and width of these CAVs are 4 m and 2 m, respectively. It is preferred for these CAVs to pass the intersection with the speed of 2 m/s and least time delay. The collisions among these CAVs are prohibited. It can be found that there are potential collisions among these CAVs. Thus, these CAVs should be coordinated to pass the intersection by the IMU. The designed VPs based on the simplified intersection are shown in this figure as well. These VPs are the same as those shown in Figure 12. Thus, the VBs in Figure 12 are used to demonstrate the initial idea.

Figure 13. Simplified intersection.

To show the initial ideal in a simple way, the dynamic limits of these CAVs are ignored. Thus, when a CAV is assigned to a VG, the CAV can reach the position of the VG with neglectable time. It is assumed that a CAV can keep a safe clearance with the front CAV in the same lane. These simplifications are made to show the initial idea of VB algorithm clearly. The VB algorithm considering the dynamic limits of CAVs will be designed next section.

Several definitions are made for VG to simplify the discussion:

- A VG that approaches to the intersection is called a **closing VG**.
- A VG that has been assigned with a CAV is called an **occupied VG**.
- A VG without CAV is called a **free VG**. A occupied VG becomes free at the time of born.
- A VG that may transfer a CAV cross the intersection with collision is called an **unsafe VG**.

Naturally, a collision avoidance policy can be proposed to prevent collisions among these CAVs: only the VGs before collision zone can be used for the transfer of CAVs. When a VG is occupied, the CVGs of the VG should be marked to avoid assignment of CAVs. Note that the mark should be cleared in certain conditions. Or a marked VG will not be utilized for CAVs forever, even all of its CVGs become free. To prevent such a condition, the mark will be cleared when the VG is dead. When a VG is born, it should be checked if there are potential conflict VGs. If a conflict VG is found, the VG should be marked to avoid usage. An unmarked VG before collision zones is called a **candidate VG**.

In order to avoid collision among CAVs, a policy is used to coordinate approaching CAVs: only a candidate VG can be used for the transfer of CAVs. The candidate VGs are ordered according to their distances to the intersection. The closer the distance, the higher order the VG. To transfer a CAV with the least time, it is preferred to assign a VG with the candidate VG with higher order.
The steps to control CAVs to pass the intersection are shown below:

Step 1: At the time of 0 s, as shown in Figure 14, find the VG \(G_{v9}^{B1}\) with the highest order in \(B_{v1}\) for the first waiting CAV. Then, assign the first waiting CAV to \(G_{v9}^{B1}\). Finally, mark the CVGs of \(G_{v9}^{B1}\) and fill them with gray.

Figure 14. Step 1, (a) find the optimal VG, (b) assign the first waiting CAV, (c) mark the CVGs.

When CAV 1 is assigned to \(G_{v9}^{B1}\), CAV 2 becomes the first waiting CAV. According to the assumption at the beginning of the section, the dynamic limits of CAVs are ignored in this section. It is assumed that a CAV can catch up with the assigned VG in no time.

Similar to step 1, the other CAVs can be assigned to the optimal VGs with steps 2–6 as shown in Figure 15.

When all CAVs have been assigned with VGs, these CAVs can get through the intersection without collision as demonstrated in Figure 16.

Figure 15. Steps 2–6, (a) step 2, (b) step 3, (c) step 4, (d) step 5, (e) step 6.
Figure 16. Transfer of CAVs, (a) $t = 3$ s, (b) $t = 6$ s, (c) $t = 9$ s, (d) $t = 12$ s, (e) $t = 15$ s, (f) $t = 18$ s, (g) $t = 21$ s, (h) $t = 24$ s, (i) $t = 27$ s, (j) $t = 30$ s.

It can be seen from the above discussion that, the designed VBs can be used for the management of intersection. The trajectories of approaching CAVs can be selected among the trajectories of these VGs. Using simple VBs, the collision between the CAVs can be avoided. Provided longer VBs, more CAVs can be transferred in a time.

5.3. Deficiencies of The Initial Ideal

If all CAVs are in the corresponding assigned VGs, there will no collision occur among these CAVs as shown in Figure 16. However, there are two deficiencies of the initial ideal, which make it difficult for practical application:

1. There are physical limits, such as maximum speed and acceleration, of CAVs. Thus, it is impractical for a CAV to catch up with the assigned VG in no time. As shown in Figure 14, in fact CAV 1 has to spend some time to catch up with the assigned VG.

2. The performance of collision avoidance between the CAVs in the same lane cannot be guaranteed. As shown in Figure 15, if CAVs 1 and 4 reach the assigned VGs simultaneously, then collisions between these CAVs may occur, which is not considered in the initial ideal.

These deficiencies can be resolved by planning a suitable trajectory considering actual physical constraints for each CAV. Following the planned trajectory, a CAV can catch up with the assigned VG and avoid collision within the physical limits. The method of trajectory planning will be discussed next section, which guarantees the feasibility of the proposed algorithm.

6. Virtual Belt Algorithm

In this section, several practical problems for the application of VB algorithm are discussed. And the deficiencies of the initial ideal mentioned previous can be resolved.

6.1. Offline Algorithm

The offline algorithm consists of VBs modeling and collision checking. Using the offline algorithm, an intersection can be modeled by several VBs. And the CVGRs of the VGs in these VBs can be obtained as well.
6.1.1. Virtual Belts Modeling

There are several steps to construct the VBs for an intersection:

1. Design the length of a VG. To ensure the safety of CAVs, a VG should be larger than CAVs with certain safe margin. Thus, the length of a VG should be longer than that of CAVs. The length of a VG is calculated as

\[ l_g = l_{veh} \cdot l_s, \]  

where \( l_s > 1 \) is a safe factor. \( l_{veh} \) is the maximum length of the considered CAVs.

2. Construct VPs based on the path of CAVs, which can be done by selecting the start points and end points of VPs. There are several aspects should be considered:

   - The start points of VPs should be selected before the collision zones in the considered intersection. The end points of VPs should be selected after the collision zones in the considered intersection. Thus, the VBs algorithm can be applied to avoid collision.
   - The lengths of these VPs should be equal.
   - If all CAVs have past the collision zone, there is no collision during the routes they drive away from the intersection with the same constant speed. Thus, it is unnecessary to place the end points far away from the intersection.
   - The VPs before the cross points should be long enough to provide sufficient VGs for the approaching CAVs. In fact this is also limited by the length of road.

3. Construct VBs based on the VPs designed previous. Each VP should be utilized to construct a VB with the same lane width. The constructed VGs in all VBs should be uniform as well.

Consider the paths of CAVs shown in Figure 3. The VPs of the intersection are designed and plotted in Figure 17. Based on these VPs, the VBs at road S can be constructed, which are illustrated in Figure 18. The VBs at roads W, E and N can be designed with the same method.

![Figure 17. VPs at intersection.](image-url)
6.1.2. Collision Checking

As these VGs constructed previous are uniform, the CVGR of each VG can be obtained by checking the collisions during a time circle of these VGs as demonstrated in Algorithm 2.

6.2. Online Algorithm

The online algorithm can be used recursively for the management of intersection, which includes the methods of VG selecting and trajectory planning.

6.2.1. Virtual Grid Selecting

As the designed VG is bigger than the CAVs, if each CAV is assigned with a candidate VG, these CAVs can pass the intersection safely. The VG selecting method for each approaching CAV can be described straightforward:

1. Find all candidate VGs suitable for the first waiting CAV.
2. Sort these candidate VGs according to their distances to the intersection.

These candidate VGs are moving to the intersection. Thus, a waiting CAV should catch up with the corresponding candidate VG before the conflict points of the intersection.

6.2.2. Trajectory Planning

As discussed in previous section, two issues should be addressed: First, a CAV should catch up with the assigned VG within a certain time. Second, the collisions among the CAVs in the same lane should be avoided. These issues should be considered for trajectory planning. Due to physical limits, a CAV may fail to catch up with the candidate VG with the highest order. On this condition, the other candidate VGs should be examined according to their orders to find the suitable one. Two assumptions are made for the algorithm of trajectory planning: First, the speed of a CAV is nonnegative. Second, a CAV without an assigned VG can maintain a safe clearance with the front CAV in the same lane.

The problem of trajectory planning is demonstrated in Figure 19. There are three VGs in a VB and two CAVs that move along the same straight lane. The speeds of these VGs are $v_c$. CAV A is the first waiting CAV. CAV B is a passing CAV and has been assigned to VG 3. The trajectory of CAV B has
been planned completely. VG 1 is a candidate VG for CAV A. \( s_c \) means the longitudinal position just before the conflict point. \( s_e \) means the longitudinal position of the VG center when the VG reaches to the conflict point. Collisions may occur if the CAV cannot catch up with the assigned VG. At the time of \( t_s \), the initial speed and position of CAV A are \( v_s \) and \( s_s \), respectively.

Figure 19. Problem of trajectory planning.

The aim of trajectory planning is to find a suitable trajectory for a waiting CAV to catch up with the candidate VG. The trajectory should be within the dynamic limits of the waiting CAV and avoid collisions with the frontal CAV. Considering these requirements, the trajectory of the CAV is planned using a polynomial based method.

The longitudinal trajectory of a CAV can be expressed as

\[
s = f(t) . \tag{4}
\]

There are mainly four constraints for the speeds and the positions of the CAV:

\[
\begin{align*}
\dot{f}(t_s) &= v_s \\
f(t_s) &= s_s \\
\dot{f}(t_e) &= v_e \\
f(t_e) &= s_e
\end{align*}
\]  

\tag{5}

where \( t_s \) and \( t_e \) are the times at the beginning and end of planned trajectory, respectively. \( v_s \) and \( x_s \) are the speed and position of the CAV at \( t_s \), respectively. \( v_e \) and \( x_e \) are the speed and position at \( t_e \), respectively.

To simplify the problem of trajectory planning, the third order polynomial is used to design the trajectory of the CAV

\[
f(t) = q_3 \cdot t^3 + q_2 \cdot t^2 + q_1 \cdot t + q_0,
\]  

\tag{6}

where \( q_{i,i=0,1,2,3} \) denote the parameters of the planned trajectory. There are four parameters need to be obtained. And there are four constraints of the speed and position of the CAV. Substituting Equation (6) into Equation (5), one has

\[
\begin{bmatrix}
3 \cdot t_s^2 & 2 \cdot t_s & 1 & 0 \\
t_s^3 & t_s^2 & t_s & 1 \\
3 \cdot t_e^2 & 2 \cdot t_e & 1 & 0 \\
t_e^3 & t_e^2 & t_e & 1 \\
\end{bmatrix}
\begin{bmatrix}
q_3 \\
q_2 \\
q_1 \\
q_0 \\
\end{bmatrix}
=
\begin{bmatrix}
v_s \\
x_s \\
v_e \\
x_e \\
\end{bmatrix}.
\]  

\tag{7}

The parameters in Equation (7) can be calculated as
Thus, taking derivation of Equation (6), the speed of the CAV can be expressed as

$$ v(t) = 3 \cdot q_3 \cdot t^2 + 2 \cdot q_2 \cdot t + q_1. $$

(9)

Taking derivation of Equation (9), the acceleration of the CAV can be expressed as

$$ a(t) = 6 \cdot q_3 \cdot t + 2 \cdot q_2. $$

(10)

It can be found from Equation (10) that the acceleration of the vehicle is a linear polynomial. Thus, $a(t_s)$ and $a(t_e)$ are the extreme values of acceleration. To limit the amplitude of acceleration, the following equations should be satisfied

$$ \begin{cases} 
  a_{\min} \leq a(t_s) \leq a_{\max}, \\
  a_{\min} \leq a(t_e) \leq a_{\max}, 
\end{cases} $$

(11)

where $a_{\min}$ and $a_{\max}$ are the lower and upper bounds of the acceleration of the CAV, respectively.

As the speed of the CAV is a quadratic polynomial, there are three possible extreme values: $v(t_s)$, $v(t_e)$ and $v(t_z)$. $t_z$ is the time which satisfies $\dot{v}(t_z) = 0$. $t_z = -\frac{q_2}{2q_3}$ can be obtained by solving

$$ \frac{v(t)}{c t} = 0. $$

(12)

If $t_s < t_z < t_e$, the maximum and minimum speeds of the planned trajectory can be expressed as

$$ \begin{cases} 
  v_{p,max} = \max(v(t_s), v(t_z), v(t_e)), \\
  v_{p,min} = \min(v(t_s), v(t_z), v(t_e)), 
\end{cases} $$

(13)

If $t_z \leq t_s$ or $t_z \leq t_e$, the maximum and minimum speeds of the planned trajectory can be expressed as

$$ \begin{cases} 
  v_{p,max} = \max(v(t_s), v(t_e)), \\
  v_{p,min} = \min(v(t_s), v(t_e)). 
\end{cases} $$

(14)

To limit the speed of the planned trajectory, the following equation should be satisfied

$$ \begin{cases} 
  v_{p,max} \leq v_{\max}, \\
  v_{p,min} \leq v_{p,min}. 
\end{cases} $$

(15)

To avoid rear-end collisions with the front CAV, the following equation should be satisfied

$$ s_f(t_i) - s(t_i) \geq d_s, $$

(16)

where $d_s$ is the minimum distance to avoid rear-end collisions. $s_f$ is the longitudinal position of the front CAV. $t_{i,1,2,\ldots,n_c}$ is the time used to check potential rear-end collisions. $n_c = t_s + i \cdot t_\Delta$ is the total count of collision checking time. $t_\Delta$ is a small time interval.

By solving Equation (8), $q_{i,0,1,2,3}$ can be obtained and a trajectory described by Equation (6) can be determined. If the trajectory is within the constraints shown in Equations (11), (15) and (16), the trajectory can be used for the CAV to catch up with the corresponding candidate VG. The trajectory may be not satisfied with Equations (11), (15) and (16) simultaneously, which means the CAV is not
able to track the selected candidate VG. And another candidate VG should be tested for that CAV using the trajectory planning method.

In light of the priority-based trajectory planning [26], when planning the trajectory of a CAV in a straight lane, only the front CAV should be considered to avoid rear-end collisions. There is no need to plan the trajectories of CAVs in the same lane simultaneously. Using the trajectory planning method, the trajectories of waiting CAVs can be planned successively based on the first-in-first-service policy.

7. Numerical Simulation

In this section, two simulations are conducted to verify the effectiveness of the proposed VB algorithm. The layout of the center of the considered intersection is shown in Figure 20. In the first simulation, several scenarios with low traffic load are constructed to compare the performances of the VB algorithm, Multi-Agent (MA) algorithm [1] and four-phase traffic lights with different green light time. In the second simulation, several scenarios with high traffic load are constructed to further compare the performances of these algorithms.

The VB and MA algorithms are performed using Matlab. Using the offline algorithm of the VB algorithm, the VBs designed at the intersection are illustrated in Figure 21. The simulations of traffic lights are conducted using VISSIM. The parameters of the intersection and CAVs in these simulations are listed in Table 1. A camera shot of the intersection controlled by the proposed VB algorithm is shown in Figure 22. The proposed VB algorithm consists of an offline algorithm and an online algorithm. The offline algorithm is used to model the considered intersection using several VBs and calculate the collision among the VGs in these VBs. As the collision of each VG with all VGs in other VBs will be checked according to Algorithms 1 and 2, the complexity of the offline algorithm is $O(C_g \cdot C_v \cdot (C_g - 1)) \approx O((C_g \cdot C_v)^2)$. Where $C_v$ and $C_g$ mean the count of VBs and the count of VGs in each VB, respectively. The online algorithm is used to calculate the trajectories of waiting vehicles in each circle one after another. The trajectory of a waiting vehicle can be obtained by successively checking the trajectories which are planned based on candidate VGs. As the candidate VGs of each vehicle is no more than $C_g$, the complexity of the online algorithm is $O(C_g \cdot C_{veh})$. Where $C_{veh}$ is the count of the waiting vehicles in each control circle. In the simulation, the computation time of the offline algorithm is about one hour. As the offline algorithm can be used before the real-time application of the VB algorithm, the time cost in the offline algorithm has no effect to the real-time performance of the VB algorithm. For the online algorithm, the time used to plan the trajectory of a waiting vehicle is less than 1 ms, which indicates that the proposed VB algorithm can be used for real-time application.

Figure 20. Center of the considered intersection.
Figure 21. VBs of intersection, (a) VBs at center, (b) VBs on road W.

Table 1. Parameters of intersection and CAVs.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lane width</td>
<td>3.75 m</td>
</tr>
<tr>
<td>Desired speed</td>
<td>36 km/h</td>
</tr>
<tr>
<td>Length of virtual belt</td>
<td>880 m</td>
</tr>
<tr>
<td>Length of virtual grid</td>
<td>8 m</td>
</tr>
<tr>
<td>Length of CAV</td>
<td>4.1 m~6.2 m</td>
</tr>
<tr>
<td>Width of CAV</td>
<td>1.6 m~2.1 m</td>
</tr>
<tr>
<td>Maximum velocity of CAV</td>
<td>30 m/s~35 m/s</td>
</tr>
<tr>
<td>Minimum velocity of CAV</td>
<td>0 m/s</td>
</tr>
<tr>
<td>Maximum acceleration of CAV</td>
<td>2.5 m/s²~4.5 m/s²</td>
</tr>
<tr>
<td>Minimum acceleration of CAV</td>
<td>−6 m/s²~−4 m/s²</td>
</tr>
</tbody>
</table>

Figure 22. Camera shot.
7.1. Simulation A

In this simulation, several scenarios are constructed to show the performance of the VB algorithm on low traffic load. The MA algorithm and traffic lights are used as comparison. The green times of these traffic lights are 17 s, 22 s and 27 s, which are denoted by TLA (traffic light A), TLB (traffic light B) and TLC (traffic light C), respectively. The mean traffic load of each road varies from 100 vehicle/h to 600 vehicle/h in the simulation.

Maximum travel time \( (t_{\text{max}}) \) is a commonly used index to compare the performances of different intersection management methods, which can be obtained by taking the maximum value of the travel times of all CAVs. Figure 23 shows the maximum travel times of the intersections managed by VB, MA, TLA, TLB and TLC in 1800 s. The maximum travel times of the scenarios using VB algorithm are further plotted in Figure 24. Figure 23 suggests that with the growing of traffic load, the maximum travel times of the intersection controlled by these algorithms do not show notable increasement. This figure indicates that, on low traffic load, the change of traffic load has little effect on the performances of these algorithms. The maximum travel times of the traffic lights methods are about 250 s on different traffic loads. The maximum travel times of the MA algorithm are higher than 150 s but lower than 200 s, which are better than these of the traffic lights methods. As a comparison, the maximum travel times of the intersection controlled by the VB algorithm are lower than 150 s, which are much less than these of the MA algorithm. These comparisons indicate that, compared with traffic lights and MA algorithm, the VB algorithm can increase traffic efficiency and save travel time for CAVs on low traffic load. The main reason is that the CAVs can be coordinated by the VB algorithm without waiting for the green light. Several candidate trajectories can be selected to find the optimal one for each CAV. Thus, they can pass the intersection with constant desired speed and little delay. For the traffic lights methods, the CAVs have to wait for green lights, which is low efficient and time consuming on low traffic load. For the MA algorithm, on low traffic load, there are a few conflicts among the trajectories of approaching CAVs. Most of these CAVs can pass the intersection without delay. While, when there is a conflict between the trajectories of two CAVs, one of these CAVs has to slow down and wait for permission next control circle. Thus, the MA algorithm shows better performance than the traffic light but worse than the VB algorithm on low traffic load.

![Figure 23. Maximum travel time of the intersection on low traffic load.](image1)

![Figure 24. Maximum travel time of the intersection controlled by the VB algorithm on low traffic load.](image2)
7.2. Simulation B

In this simulation, several scenarios are constructed to show the performance of the VB algorithm on high traffic load. The MA algorithm and traffic lights are used as comparison. The mean traffic load of each road varies from 500 vehicle/h to 3000 vehicle/h in the simulation.

Figure 25 shows the maximum travel times of the intersections managed by VB, MA, TLA, TLB and TLC in 1800 s. The maximum travel times of the scenarios using VB algorithm are further plotted in Figure 26. Figure 25 suggests that with the growing of traffic load, the maximum travel times of the intersection controlled by TLA, TLB and TLC increase remarkably. The maximum travel times exceed 500 s on the traffic load of 2000 vehicle/h. The MA algorithm shows better than the TL algorithm just on the traffic load of 500 vehicle/h and 1000 vehicle/h. The maximum travel time shows a notable increase on the traffic load of 1500 vehicle/h, which is worse than the traffic lights methods. This is because on high traffic load, there are more conflicts among the trajectories of approaching CAVs. Most of these CAVs with conflicts will stop before the collision zone, which leads to much loss of time. As a comparison, the maximum travel time of the intersection controlled by the VB algorithm shows less increasement than other algorithms, which is below 170 s as indicated in Figures 25 and 26. The main reason is that, on high traffic load, the CAVs can be coordinated by the VB algorithm using trajectory planning to avoid collision. And less CAVs need to wait before the intersection. Thus, the traffic efficiency can be improved than the compared algorithms. These comparisons indicate that the VB algorithm can increase traffic efficiency and save travel time greatly for CAVs on high traffic load.

8. Conclusions

This study proposes VB algorithm for intersection management to improve traffic efficiency of intersections. The proposed VB algorithm not only can be used for commonly discussed intersection, but also provides a framework to the investigation of general intersection management. As the VBs can be constructed based on the vehicle paths, the offline algorithm can also be applied for roundabout and other types of intersections without modification. Moreover, as indicated in the considered intersection, the overlap among VBs is also acceptable for the VB algorithm. Thus, the offline algorithm provides a flexible way to model an intersection. Because the calculation of the offline algorithm can be done
before real-time application, the complexity of the intersection will not affect the real-time performance of the VB algorithm. When an intersection is modeled as several VBs, the online algorithm can be utilized to plan the trajectories of waiting vehicle to pass the intersection. The core of the online algorithm is that, based on the constructed VBs, several candidate trajectories can be selected for each approaching CAV. Thus, there are more opportunities can be provided by the VB algorithm for an approaching CAV through the intersection. Since the constraints of the speed and acceleration of CAVs are considered, the online algorithm can be used for different kinds of vehicles on road, which guarantees the flexible of the VB algorithm to common vehicles. In the simulation, instead of a homogeneous intersection with a symmetrical structure, an inhomogeneous intersection with a general structure is considered to represent a general case. The counts of VBs in different roads are not the same and there are overlaps among these VBs. Moreover, the considered CAVs are of different dynamic performances, such as maximum speed and maximum acceleration. The successful application of the proposed VB algorithm indicates that this algorithm is effective to manage such a general intersection.

There are three research directions for the future investigation: The first one is to apply the VB algorithm to other types of intersection, such as roundabout and 3-way intersection. The second one is to extend the VB algorithm to the management of multi-intersection. The third one is to investigate the VB algorithm with the fuel economy performance.

Author Contributions: Conceptualization, C.B. and G.Y.; methodology, C.B.; validation, C.B., G.Y. and L.X.; writing—original draft preparation, C.B.; writing—review and editing, N.Z.; supervision, G.Y.

Funding: This research was funded by National Natural Science Foundation of China, grant number 51575103, U1664258, National Key R&D Program in China, grant number 2016YFB0100906, 2016YFD0700905, Foundation of State Key Laboratory of Automotive Simulation and Control, grant number 20160112, Scientific Research Foundation of Graduate School of Southeast University and Southeast University Excellent Doctor Degree Thesis Training Fund, grant number YBJJ1703, YBJJ1704.

Acknowledgments: The authors wish to express their gratitude to the anonymous reviewers and the associate editor for their rigorous comments during the review process.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations
The following abbreviations are used in this manuscript:

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAV</td>
<td>Connected and Autonomous Vehicle</td>
</tr>
<tr>
<td>CVG</td>
<td>Collided Virtual Grid</td>
</tr>
<tr>
<td>CVGR</td>
<td>Collided Virtual Grid Record</td>
</tr>
<tr>
<td>DSRC</td>
<td>Dedicated Short Range Communication</td>
</tr>
<tr>
<td>MA</td>
<td>Multi-Agent</td>
</tr>
<tr>
<td>IMU</td>
<td>Intersection Management Unit</td>
</tr>
<tr>
<td>RCAV</td>
<td>Request of a CAV</td>
</tr>
<tr>
<td>TLA</td>
<td>Traffic Light with the green time of 17 s</td>
</tr>
<tr>
<td>TLB</td>
<td>Traffic Light with the green time of 22 s</td>
</tr>
<tr>
<td>TLC</td>
<td>Traffic Light with the green time of 27 s</td>
</tr>
<tr>
<td>VB</td>
<td>Virtual Belt</td>
</tr>
<tr>
<td>VG</td>
<td>Virtual Grid</td>
</tr>
<tr>
<td>VP</td>
<td>Virtual Path</td>
</tr>
</tbody>
</table>

References


© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).