2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Decentralized Near-to-Near Approach for Vehicle Platooning based on Memorization and Heuristic Search

Jano Yazbeck, Alexis Scheuer and François Charpillet
Université de Lorraine, LORIA
Inria, MAIA team - FRANCE
[Jano.Yazbeck | Alexis.Scheuer | Francois.Charpillet]@loria.fr

***Abstract*— This paper deals with vehicle platooning, where a convoy aims at following closely and safely its leader's path without collision nor lateral deviation. In this paper, we propose a platooning algorithm based on a near-to-near decentralized approach. Each vehicle estimates and memorizes on-line the path of its predecessor as a set of points. After choosing a suitable position to aim for, the follower estimates on-line the predecessor's path curvature around the selected target. Then, based on a heuristic search, it computes an angular velocity using the estimated curvature. The optimization criteria used in this work allows the robot to follow its predecessor's path without oscillation while reducing the lateral and angular errors.**

I. INTRODUCTION

In the European project InTraDE (http://www.intrade-nwe.eu), we aim at automating the maritime ports by introducing electrical autonomous vehicles to transport the containers efficiently. Moving in an unstructured environment, each human operator moves containers along a convenient path from the discharge zone to the storage area. In order to move the containers more efficiently, we propose to form a convoy of robu-tainers (specific vehicles built by Robosoft) leaded by the a single driver. Using the platooning technique, the convoy moves along a path with no previous knowledge of its analytic equation.

Different approaches have been proposed for platooning: centralized, global decentralized and local decentralized approaches.

- In **centralized approaches**, a central controller collects data from all the vehicles. Then, it computes and sends the commands to each vehicle (e. g. [1], [2]). In **global decentralized approaches** (as [3], [4], [5]), each vehicle collects data relative to the state of the whole platoon and computes its own commands. The computation of the commands in centralized and global decentralized approaches relies on the state of the whole platoon. Robots achieve an accurate tracking but are not independent. Effects on the platoon must be considered in case the leading vehicle crashes or faces technical problems. They rely on communication for data exchange. Unfortunately, as discussed in [6], using communication can cause problems such as delays due to packet loss, data transmission time and their analyze's time.
- In **local decentralized approaches** ([7], [8], [9]), each vehicle perceives locally its environment and computes its behavior. Robots are fully autonomous given that they do not rely on communicated data to compute their commands. However, the main drawback in this kind of approaches is the propagation of an error, along the platoon, induced by the local perception and accumulated from vehicle to vehicle.

This paper presents a platooning algorithm where the convoy aims to reproduce precisely its leader's path under a decentralized local approach. The approach proposed in this work is based on an heuristic search. Given that the leader's path is unknown to the following robots, each robot perceives its predecessor via its embedded sensors and aims at following its path. In a previous paper [10], we presented the $Memo-LAT$ algorithm (for path Memorization and Look-Ahead Target selection), a decentralized local algorithm based on memorizing and tracking the predecessor's path instead of its current position. Even if this method drastically reduces the lateral deviation, the precision and the stability of the platooning is based on the given value of a certain parameter d_l. This paper presents NOC, an improvement of the $Memo-LAT$ algorithm: in order to avoid oscillations, the angular velocity is computed while considering the path's curvature around the selected target. This curvature is estimated on-line during the robot's movement, and the computation of the angular velocity is based on an heuristic search. This NOC algorithm (for Non-Oscillatory Convergence) does not need parameter adjustment.

This paper is organized as follows. First, Section II presents the framework. Then, Section III presents the proposed algorithm for platooning. In Section V, we explain the on-line approximation of the predecessor's path curvature during the robot's movement. The lateral control law conception is then detailed in Section VI, and numerical simulations are shown in Section VII to validate our algorithm. Finally, we conclude in Section VIII.

II. FRAMEWORK

In maritime ports, electric vehicles move at a low velocity ($v \leq 30\ km.h^{-1}$) for safety reasons. In this paper, the convoy is moving with bounded longitudinal and angular velocities $v_{min} \leq v \leq v_{max}$ and $\omega_{min} \leq \omega \leq \omega_{max}$ and with a bounded longitudinal acceleration $a_{min} \leq a \leq a_{max}$. We

978-1-4799-3684-7/14/$31.00 ©2014 IEEE

consider a mobile robot whose dynamics can be represented by the following equations:

$$\begin{cases} \dot{x} = v.\cos\theta \\ \dot{y} = v.\sin\theta \\ \dot{\theta} = \omega, \qquad \omega_{min} \leq \omega \leq \omega_{max} \\ \dot{v} = a, \qquad a_{min} \leq a \leq a_{max} \end{cases} \tag{1}$$

where (x, y) are the Cartesian coordinates of the robot, θ its orientation and (v, ω) its longitudinal and angular velocities. The commands applied on the robot at each time step are the longitudinal acceleration a and the angular velocity ω.

III. PROPOSED ALGORITHM

Given that our approach is a near-to-near decentralized one, we present the platooning algorithm for a two-robot convoy (leader-follower) and then we show numerical simulations on a 7-robot convoy. The NOC algorithm is summarized as follows:

- acquire and store the predecessor's new position ;
- compute the longitudinal acceleration a which avoids collision with the preceding robot (Sec. IV) ;
- choose, among the stored positions, the best one to aim for (see the explanation in the paragraph below) ; the chosen position is called the target ;
- consider the memorized path around the target to estimate its curvature (Sec. V) ;
- compute, using the curvature, the angular velocity which reduces the lateral and angular errors of the follower without crossing the approximated path (Sec. VI).

The controller does not choose a target located at a constant look-ahead distance (corresponding to the path's curvature) as in the $Memo - LAT$ algorithm [10]. On the contrary, it looks for the closest position of its predecessor's path for which a valid angular velocity can be found. To do so, the controller checks for each stored point of its predecessor's path (and starting from the last aimed one) if the robot can turn with its maximum angular velocity without crossing the approximated path around this point. If no intersection occurs, the goal is found.

IV. LONGITUDINAL CONTROL

We use in this work a longitudinal controller [11] which allows a short inter-distance platooning while ensuring collision avoidance. A pair of successive robots maintain a secure inter-distance even in critical cases where the leader stops.

The follower computes the suitable longitudinal acceleration based on its current velocity v, the velocity of its predecessor v_p, and other values such as the minimum allowed inter-distance d_{crit} and the maximum acceleration a_{max} and deceleration a_{min} fixed by the vehicle dynamics.

To estimate the predecessor's velocity v_p, the controller computes Δd, the variation of the inter-distance between the two successive robots during a time step Δt. Then, it estimates the preceding robot velocity as $v_p = v + \Delta d/\Delta t$.

Once the preceding vehicle velocity is computed, the controller computes a bounded longitudinal acceleration $(a_{min} \leq a \leq a_{max})$ using the formula given in [11, (2)]. The new linear velocity is then computed from the previous velocity v_0 using the following equation:

$$v = a.t + v_0, \quad v_{min} \leq v \leq v_{max} \tag{2}$$

V. ON-LINE PATH APPROXIMATION

In order to deduce the curvature of the leader's path at the selected target, the follower approximates the corresponding portion of the path by an arc of a circle or a line segment. We distinguish two cases:

- the target is the first point of the preceding robot's path: in this case, the approximated path is the line joining the follower's initial position with its predecessor's initial position. When the distance to its predecessor's initial position becomes less than the distance traveled during a time step $(v.\Delta t)$, and if the follower has acquired at least two positions of its predecessor, the approximated path becomes the line joining the first two points of the predecessor's path.
- the target is not the first position of the predecessor's path: the lateral controller picks the position before and the one after the target. Then, if these three points are aligned, it constructs the linear path joining them. Otherwise, the approximated path is the circle generated by these three positions. If the target is the current position of the preceding robot, the follower does not have a position after this target. In this case, the lateral controller considers the last three positions in its memory and constructs the line or the circle joining them. If the follower did not acquire more than two positions from its predecessor's path, the approximated path is the line joining these two points.

VI. LATERAL CONTROL

In this paper, we do not consider an analytic expression of the control law. Instead, we build a controller which discretizes the angular velocity domain $[\omega_{min}, \omega_{max}]$ and chooses the best commands to guarantee a non-oscillatory platooning.

We consider a robot initially located at (x_0, y_0) and oriented with θ_0, moving at a constant velocity v_0. If we apply an angular velocity ω_i $(\omega_i \neq 0)$, the robot follows a circular arc $(\mathcal{C}_i)$ centered at $\Omega_i(x_0 - v_0.\sin\theta_0/\omega_i,\ y_0 + v_0.\cos\theta_0/\omega_i)$ and with a radius $R_i = |v_0/\omega_i|$. If $\omega_i = 0$, the robot generates a line segment $[\mathcal{D}_i]$. Starting at $(x_0,\ y_0,\ \theta_0)$, the robot arrives at $q_i(x_i,\ y_i,\ \theta_i)$ after a time step Δt.

After approximating the predecessor's path, the robot computes a sequence of angular velocities to follow the approximated path. In order to avoid oscillations generated by angular errors, the follower aims at converging towards the approximated path with an orientation tangent to this path. As we try to minimize both the lateral deviation and the angular error (shown in Fig. 1), the optimization criteria is therefore an error function $\mathcal{E}$ of these two variables.

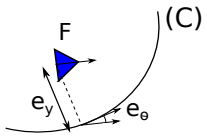

Fig. 1. Lateral ϵ_y and angular ϵ_θ errors for the optimization criteria.

A. Platoon Initialization

To avoid crossing the leader's path, the follower should be sufficiently far from this path so that it can turn with its maximum angular velocity without intersecting it. Thus, the radius of the circle $(\mathcal{C}_{max})$ generated by applying ω_{max} on the robot must be smaller than (or equal to) the distance from its center $\Omega_{max}(x_0 - v_0.\sin\theta_0/\omega_{max},\ y_0 + v_0.\cos\theta_0/\omega_{max})$ to its target path $(\mathcal{C})$ (or (Δ)) called d_{max} (Figure 2).
To avoid crossing the predecessor's path, the robot should be initially positioned so that d_{max} verify the following relation:

$$d_{max} \geqslant \frac{v_0}{\omega_{max}} \tag{3}$$

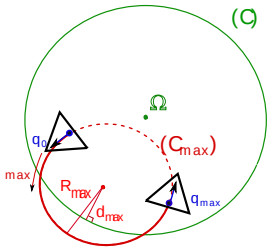

Fig. 2. Platoon's initialization in the case of a circular leader's path.

If the follower changes its target path (because it chooses another target, as explained in Section V), the initialization condition may not be verified. This can lead the follower to cross its predecessor's path before correcting its orientation and converging towards it.

B. Angular Velocity Computation

From the discretized domain $\{\omega_{min}, ..., \omega_{max}\}$, the lateral controller retains the angular velocities ω_i which, by generating the corresponding path ($(\mathcal{C}_i)$ or $[\mathcal{D}_i]$), verifies the two following conditions:

- Condition Γ_i: the arc of a circle $(\mathcal{C}_i)$ (or the line segment $[\mathcal{D}_i]$) and $(\mathcal{C})$ (or (Δ)) does not intersect during Δt.
- Condition Γ_i^+: starting from the possible new configuration q_i, the entire circle generated by the follower turning with a maximum angular velocity (ω_{max} or ω_{min}) does not cross the approximated path $(\mathcal{C})$ (or (Δ)).

Then, from the retained angular velocities, the lateral controller chooses the one which minimizes the predefined error function $\mathcal{E}$.

By verifying these two conditions (Γ_i and Γ_i^+), the controller anticipates future intersections and corrects its orientation. As we can see in Fig. 3, if ω_i is the chosen command, the robot can avoid crossing its target path by turning with an angular velocity equal to ω_{max}.

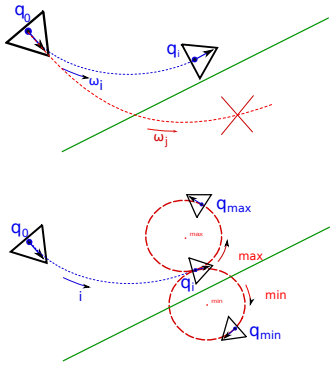

Fig. 3. Verification of the two conditions: Γ_i (top) and Γ_i^+ (bottom).

C. Optimization criteria

The purpose of using the heuristic search in this work is to choose an angular velocity from the discretized domain $\{\omega_{min}, ..., \omega_{max}\}$ which reduces the most the lateral error ϵ_y while considering the angular error ϵ_θ.

For each ω_i of the retained angular velocities (which verify the conditions Γ_i and Γ_i^+), we proceed as follows (see Fig. 4):

- starting from q_0, compute the new configuration q_i if ω_i was applied on the follower ;
- at q_i, compute the lateral ϵ_y and angular ϵ_θ errors relative to the approximated path (Δ) or $(\mathcal{C})$;
- if ϵ_θ is equal to zero, the error function $\mathcal{E}$ is equal to ϵ_y. Otherwise, starting from q_i, compute the new position q_i^+ of the follower obtained by turning, during several time steps, with the maximum angular velocity and where the angular error relative to (Δ) or $(\mathcal{C})$ becomes equal to zero. At q_i^+, the error function $\mathcal{E}$ is equal to the lateral error relative to (Δ) or $(\mathcal{C})$.

So, by computing the lateral error at q_i^+, when $\epsilon_\theta = 0$, instead of q_i, we take the influence of the angular error into account directly in the expression of the lateral error.

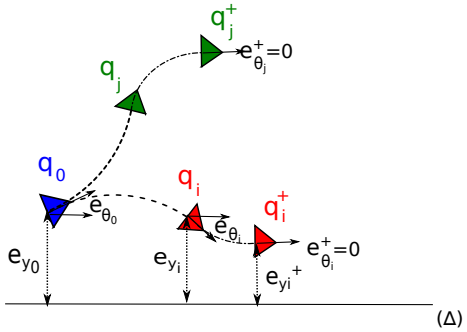

Fig. 4. Optimization criteria steps for a linear path (Δ).

Fig. 4 shows two examples of possible commands: ω_i and ω_j. If the follower, initially at q_0, applies ω_i, it will arrive

at q_i. At q_i, the follower will turn with ω_{max} during several time steps until it reaches a position q_i^+ where its orientation is parallel to the approximated path (Δ). The error function $\mathcal{E}_i$ is, in this case, the lateral error of the follower at q_i^+ relative to (Δ).

If the follower applies ω_j at q_0, it will arrives at q_j. Then, it will turn with $-\omega_{max}$ in order to reach q_j^+ where its orientation is parallel to (Δ) and where the error function $\mathcal{E}_j$ is equal to the lateral deviation at q_j^+.

By comparing $\mathcal{E}_i$ and $\mathcal{E}_j$, the controller chooses the command ω_i since the lateral deviation $\epsilon_{y\,i}^+$ is smaller than $\epsilon_{y\,j}^+$.

The same reasoning can be applied in case the predecessor's path is approximated by a circle.

D. Refinement of the Angular Velocity

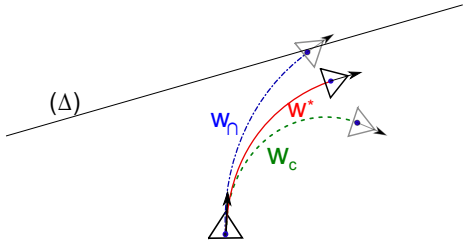

Fig. 5. The new domain for the angular velocity refinement.

By proceeding as explained in Subsection VI-B, the lateral controller chooses ω_c as the best command among the possible values. However, the chosen angular velocity may not be the optimal command: while testing all the possible commands in the set $\{\omega_{min}, ..., \omega_{max}\}$, the controller might find an angular velocity $\omega_\cap$ which satisfies only Γ_i (Γ_i^+ is not verified as the robot crosses its predecessor's path while turning with ω_{max} at the next time step). In this case, the controller can find in the domain $[\omega_\cap, \omega_c]$ a command ω^* better than ω_c which verifies both Γ_i and Γ_i^+ and reduces the error function more than ω_c. In order to compute ω^*, the controller discretizes $[\omega_\cap, \omega_c]$ into N values and proceeds as explained in Subsection VI-B.

E. Combination of the Longitudinal and Lateral Controllers

During the elimination process of the angular velocities which lead to oscillations around the leader's path, we supposed previously that the follower is moving at a constant velocity. Yet, this is not the case due to the longitudinal control, which maintains a safe secure distance with its predecessor. In this case, the path followed during a time step Δt by the follower moving with ω_i and v ($v = a.t + v_0$ and $0 \leq t \leq \Delta t$) is not a circle, but an arc of a spiral. This portion of the spiral can be bounded by two circles, one generated by (v_0, ω_i) and the other by $(a.\Delta t + v_0, \omega_i)$ as shown in Fig. 6. In order to ensure that all the retained angular velocities prevent the follower from intersecting the approximated path, we consider in the angular velocity computation (see Section VI-B) the circle with the largest radius. In other terms, $(\mathcal{C}_i)$ is the circle corresponding to $v_i = \max(v_0, a.\Delta t + v_0)$, i.e., the red circle in Fig. 6.

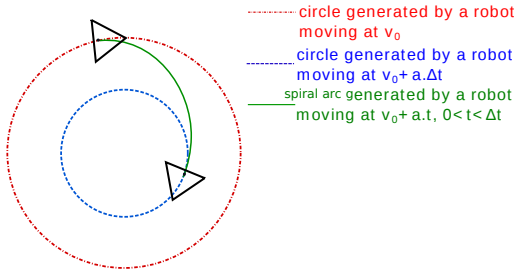

Fig. 6. A spiral arc and its circular bounds.

F. Stability of the Platooning Algorithm: does the follower always converge towards its leader's path?

Proving the stability of the proposed platooning algorithm consists in proving the convergence of a robot under NOC algorithm towards its leader's path regardless its configuration. This convergence is obtained when the error to the path decreases: $|\frac{e_{i+1}}{e_i}| < 1$. Given that the leader's path is approximated by a succession of line segments and circle arcs, we have to prove the convergence of the follower towards both a linear and a circular paths. We consider a misguided robot (its orientation is not directed toward its leader's path) and we compute the number of time steps N^* needed so that the follower corrects its orientation and starts converging towards its leader's path. In some cases, one step time is not enough to correct the robot's orientation: the controller applies $\pm\omega_{max}$ to converge quickly. Note that, when the target path changes, the robot's convergence towards this new path is guarantied if this target path does not change for at least N^* time steps. As mentioned previously, during the computation of the lateral behavior of the follower, we suppose that its linear velocity is constant and equal to $\max(v_0, v_0 + a.\Delta t)$.

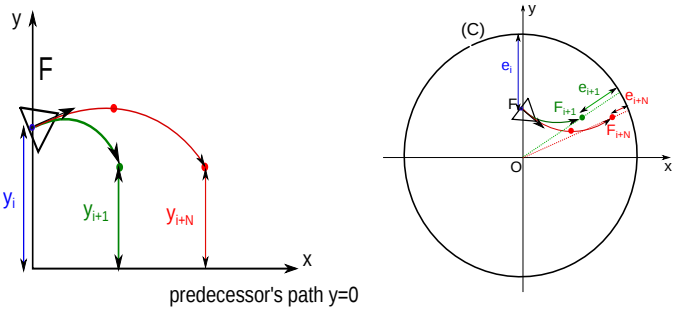

Fig. 7. Convergence to a line segment or to a circular arc.

1) Convergence to a line segment: As Figure 7 shows, we consider the coordinate system where the x-axis is the predecessor's path ($y = 0$) and where the y-axis passes through the follower ($x_f = 0$). In this case, the error of the follower to its predecessor's path is $e = y_f$. The proposed algorithm is stable if the follower converges towards its leader's path: $e_{i+1} < e_i$, or in other terms, $y_{i+1} < y_i$.

By integrating $\dot{y} = v \sin\theta$ where $\theta(t) = \omega.t + \theta_0$ and v is constant, we obtain the following equation:

$$y_{i+N} = y_i - \frac{v}{\omega_i}\left(\cos(N.\omega.\Delta t + \theta_i) - \cos\theta_i\right); \quad N > 0 \tag{4}$$

To satisfy $y_{i+1} < y_i$, $\frac{v}{\omega_i}\left(\cos(\omega.\Delta t + \theta_i) - \cos\theta_i\right)$ must be positive. We study the sign of the fraction $A = \frac{\cos(\omega.\Delta t + \theta_i) - \cos\theta_i}{\omega_i}$ with the hypothesis $0 \leq \theta_i \leq \frac{\pi}{2}$, given that $\frac{-\pi}{2} \leq \omega.\Delta t + \theta_i \leq \frac{\pi}{2}$ and $\omega_{min} \leq \omega_i \leq \omega_{max}$. In this case, the lateral controller computes a negative angular velocity in order to correct the robot's orientation. we can distinguish two cases:

- $|\theta_{i+1}| > |\theta_i| \Rightarrow \cos\theta_{i+1} < \cos\theta_i \Rightarrow A > 0$. When the new orientation of the follower verifies $\frac{-\pi}{2} \leq \theta_{i+1} \leq -\theta_i$, the error to the leader's path decreases, proving the stability of the algorithm.
- $|\theta_{i+1}| < |\theta_i| \Rightarrow \cos\theta_{i+1} > \cos\theta_i \Rightarrow A < 0$. In this case, the convergence is not established after one time step. The follower needs more time steps to re-directed and converge towards its leader's path. In order to obtain $A > 0$, θ_{i+N} must be less or equal to $-\theta_i$ where θ_{i+N} is the new orientation of the follower turning with the maximum angular velocity ($\omega_{min} = -\omega_{max}$) during N^* time steps:

$$\begin{aligned} \theta_{i+N} &= -N.\omega_{max}\Delta t + \theta_i \leq -\theta_i \\ N\Delta t &\geq \frac{2.\theta_i}{\omega_{max}} \; ; \; N^*\Delta t = \frac{2.\theta_i}{\omega_{max}} \end{aligned} \tag{5}$$

2) Convergence to a circular arc: To prove the stability of NOC algorithm in this case, we set the coordinate system at the center O of the circle (of radius R), as shown in Figure 7, oriented so that the robot is on the y-axis. We suppose that the robot is inside the circle and its orientation is between $-\frac{\pi}{2}$ and $\frac{\pi}{2}$. If we call d the distance of the follower to the center O, the error of the follower to the leader's path is expressed as $e = |R - d|$ (see Fig. 7). At the instant i, the distance d_i is equal to y_i.

The proposed algorithm is stable if the error e_i to the circular path decreases over time: $e_{i+1} < e_i$. This means that the distance of the follower to the circle center O increases: $d_{i+1} > d_i \Rightarrow x_{i+1}^2 + y_{i+1}^2 > y_i^2$. By integrating the equations of the dynamic model 1, where v is constant, we obtain the evolution of the robot's position after N time steps:

$$\begin{aligned} x_{i+N} &= x_i + \frac{v}{\omega_i}\left(\sin(N.\omega_i.\Delta t + \theta_i) - \sin\theta_i\right); \quad x_i = 0 \\ y_{i+N} &= y_i - \frac{v}{\omega_i}\left(\cos(N.\omega_i.\Delta t + \theta_i) - \cos\theta_i\right); \quad N \geq 0 \end{aligned} \tag{6}$$

Then, for $N = 1$, $x_{i+1}^2 + y_{i+1}^2 > y_i^2$ gives:

$$2.\frac{v_0^2}{\omega_0^2}.(1 - \cos(\omega_i\Delta t)) - 2.y_i.\frac{v_0}{\omega_i}(\cos\theta_{i+1} - \cos\theta_i) > 0 \tag{7}$$

This inequality is always satisfied except in two cases:

1) $\theta_i > 0$, $\omega_i < 0$ and $\theta_{i+1} < -\theta_i$;
2) $\theta_i < 0$, $\omega_i > 0$ and $\theta_{i+1} < -\theta_i$.

This means that after one time step, the robot does not correct its orientation. So, it turns with its maximum angular velocity during N^* time steps in order to start converging towards its leader's path. In this case, $\theta_{i+N} \geq -\theta_i$, for $N \geq N^*$:

$$\begin{aligned} N\Delta t.\omega^* + \theta_i &\geq -\theta_i \\ N\Delta t. &\geq -\frac{2.\theta_i}{\omega^*} \; ; \; N^*.\Delta t = -\frac{2.\theta_i}{\omega^*} \end{aligned} \tag{8}$$

where ω^* is equal to $\pm\omega_{max}$ depending on the direction the robot has chosen to correct its orientation: if $\theta_i > 0$, the robot will turn with $-\omega_{max}$ (case 1) ; and if $\theta_i < 0$, it will turn with ω_{max} (case 2). In both cases, after N^* time steps, the convergence is established and the stability of the proposed algorithm is proven.

In order to prove the convergence of a robot located outside the circle, a similar reasoning can be followed. The only difference is that the error is expressed differently: $e = R + d$.

VII. EXPERIMENTAL RESULTS

We consider a seven-robot convoy. The maximum longitudinal velocity is $v_{max} = 8\ m/s$. The robots are initially equidistant on a straight path (the distance between two successive robots is $0.9m$). They can turn with a maximum angular velocity $\omega_{max} = \frac{\pi}{3}\ rad/s$, so that the minimum turning radius is $8m$. The minimum allowed distance between each pair of following robots is $d_{crit} = 0.5m$.

Each follower uniformly discretizes its command domain into 10 values, and computes an angular velocity ω bounded between $\pm\omega_{max}$. Also, the new angular velocity domain for the refinement is discretized into 10 values.

A. Comparison between $Memo - LAT$ and NOC

As we mentioned in the introduction, this paper presents the NOC algorithm as an improvement of the previous algorithm $Memo - LAT$. Figure 8 shows the evolution of the lateral error between the follower's and its predecessor's paths along a spiral path. As we can see, following the reconstructed path of the predecessor's instead of tracking successive positions from the predecessor's path reduces drastically the lateral deviation.

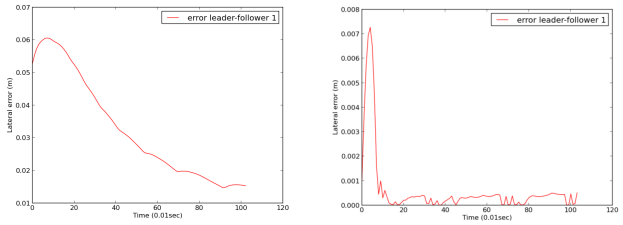

Fig. 8. $Memo - LAT$ and NOC comparison along a spiral path.

B. Stability of the platooning algorithm

In order to verify the proof of stability demonstrated in Subsection VI-F, we realize a two-robot convoy platooning along a line. We consider a follower with initial lateral and angular errors. As we can see in Figure 9, the follower

converges towards its leader's path. The stability of the platooning is proven by the decrease of the lateral error and its convergence to zero.

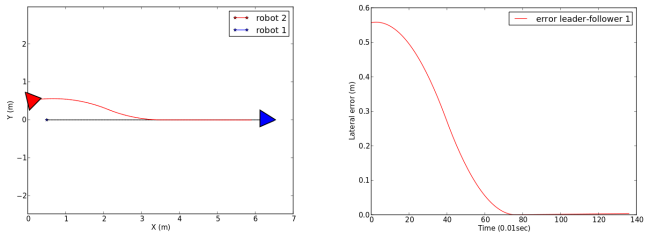

Fig. 9. Linear path stability: platooning (left), lateral error (right).

C. Platooning around a rounded corner

We consider a two-robot convoy moving along a rounded corner. The lateral deviation between the paths of the follower and the leader is represented in Fig. 10. As we can see, the error was equal to zero when the follower was moving exactly on the line of the leader. Then, after following the rounded corner with small lateral deviations, the robot reaches the linear path with a very small lateral error (less than $1cm$). As we can notice, the leader's path curvature changes roughly (from $0.067m$ to 0) when the follower is too close to its leader's path, which induces a strong angular error: the robot follows precisely its leader's path with a lateral error less than 2 cm.

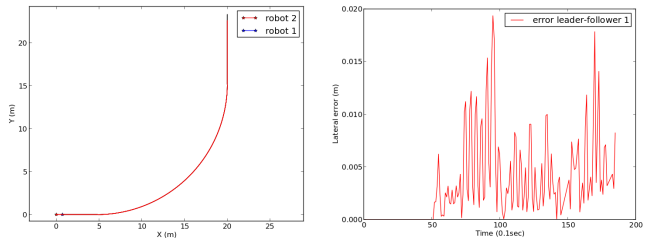

Fig. 10. left: platooning ; right: lateral error.

D. Platooning along a spiral path

Figure 11 shows a platooning of a seven-robot convoy along a spiral. The lateral deviation between the paths of each follower and the leader of the convoy is shown in Fig. 12. As we notice, even if the lateral error accumulates from robot to another, the maximum lateral error remains small ($1.3cm$).

E. Impact of the instantaneous variation of the leader's velocity on the platooning

In this section, we study the influence of the longitudinal controller on the lateral controller by varying the velocity of the convoy's leader. We consider two robots moving at a constant velocity $v = 4\ m/s$, and where the follower is at $0.9m$ away from its leader. The leader generates the following path (see Fig. 13):

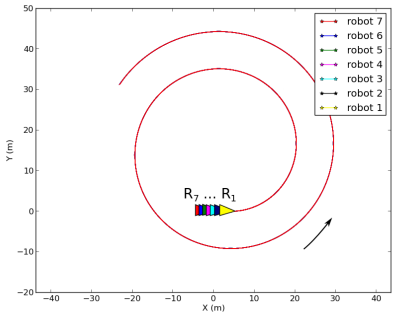

Fig. 11. Platooning of a seven-robot convoy along a spiral.

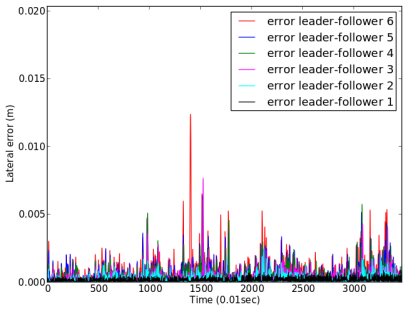

Fig. 12. Lateral deviation.

- an arc of a circle by moving at constant linear and angular velocities $v = 4m/s$, $\omega = \frac{\pi}{12}rad/s$ during $3s$.
- a segment of a line by moving with a longitudinal acceleration $|a| = 0.5m/s^2$ during $1s$.
- an arc of a circle by moving at constant linear and angular velocities $v = 3.5m/s$ (or $v = 4.5m/s$), $\omega = -\frac{\pi}{12}rad/s$ during $6s$.

In the following subsections, we study the behavior of a following robot when the longitudinal acceleration is positive or negative respectively.

1) The leader accelerates: $a = 0.5m/s^2$: As we can see in Fig. 15, the follower accelerates at the beginning of the platooning in order to reduce the initial inter-distance with its leader. Once a safe inter-distance is established, it decelerates along the arc of the circle in order to avoid collisions. As shown in Fig. 14, the maximum lateral error along this circular path is $0.04cm$.

When the follower reaches the linear path, its lateral error is very small $(0.003cm)$ (the follower is moving precisely along the circular path of its leader). This large modification of the curvature at a very small lateral error induces a lateral deviation (first peak of $0.12cm$ at $t = 3s$) because the follower reaches the linear path with a small angular error (the follower is not oriented tangentially to the linear path). Then it tries to reduce this lateral error to $0.044cm$ by applying suitable angular velocities. However, since the follower has approached very closely the linear path (generated by its leader) with a non-tangent orientation, it computes an angular

velocity which drives it away from this line for two reasons:

- among the possible commands, the controller can not find an angular velocity which reduces the angular error without increasing the lateral deviation ;
- since the velocity of the follower is growing along the linear path, the lateral controller considers the new velocity obtained by the longitudinal controller in the angular velocity computation. The higher the velocity is, the more important is the possibility of intersection. The follower tends to apply angular velocities which avoid oscillations and increase (in worst-case scenario) the lateral error.

This explains the several peaks occurring along the linear path, as shows Fig. 14.

Finally, the follower reaches the last portion of its leader's path (the arc of circle) and follows it with a maximum lateral deviation equal to $0.04cm$. As expected, the longitudinal controller allows the follower to accelerate when its leader does (Fig. 15). Also, when the leader moves at a constant velocity, the follower's velocity converges towards the same velocity.

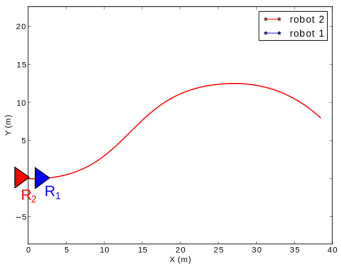

Fig. 13. Platooning along a curve under acceleration mode.

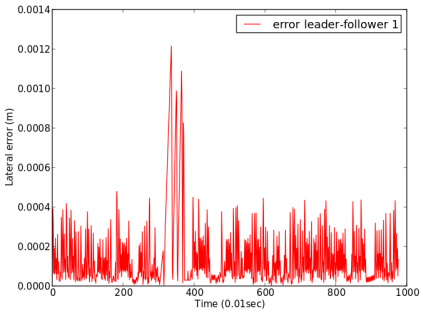

Fig. 14. Lateral deviation between the paths of the leader and its follower.

2) The leader decelerates: $a = -0.5m/s^2$: The same behavior of the follower can be observed in this case, with a slight difference along the linear path. As we can see in Fig. 16, the number of peaks along the linear path is highly reduced thanks to the decrease of the longitudinal velocity. Since its velocity is decreasing, the follower can approach its leader's path while reducing the angular and lateral errors at the same time. Then, the follower reaches the circular path with a smaller lateral deviation $(0.01cm)$ in comparison with the Subsection VII-E.1 experiment where the lateral deviation was $0.04cm$. Thus, the following robot follows this circular arc more precisely.

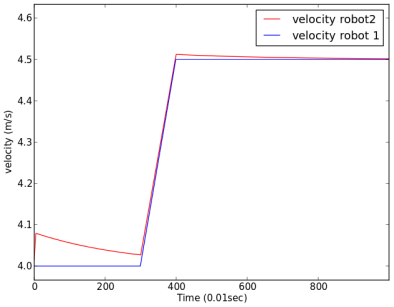

Fig. 15. Variation of the two robots' velocities during platooning.

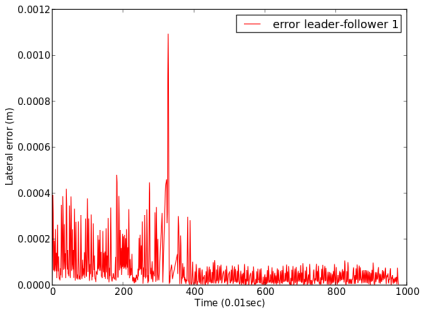

Fig. 16. Lateral deviation between the paths of the leader and its follower.

F. Impact of the instantaneous variation of the curvature on the platooning

We consider a robot R_1 moving with a constant longitudinal velocity $v = v_{max} = 8\ m/s$. Its angular velocity is varying between ω_{max} and $-\omega_{max}$ each $0.5s$. A following robot R_2 tries to follow the leader's generated path while moving at the same longitudinal velocity $v = 8\ m/s$.

Figure 17 shows the lateral error between the two successive robots' paths: after following a linear path joining its initial position with the initial position of R_1, R_2 reaches A where the curvature of the approximated path suddenly changes. Even if R_2 follows the arc AB with its maximum angular velocity, a slight delay induces an increase of the lateral error from $0.1cm$ to $3cm$. At B, the curvature of the leader's path changes sign. This sign switch allows R_2 to approach its leader's path since it is placed on the right side, and to reach it at C. Once R_2 reaches C, it approximates a linear path (the tangent at the inflection point) and keeps reducing its lateral deviation along CC'. The same phenomenon happens again with a tiny difference: the lateral error of the follower at C' $(0.005cm)$ is smaller than the one at A $(0.1cm)$ which induces a smaller lateral deviation at D $(2.25cm)$ (where the curvature changes sign) than at

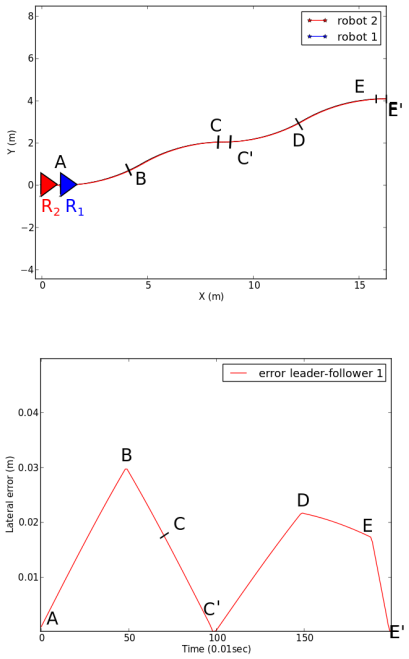

Fig. 17. top: platooning along a extremely winding path ($\pm\omega_{max}$). bottom: lateral deviation.

B $(3cm)$. The follower could not reduce this deviation along the arc DE as quickly as it did along the arc BC due to the limited choice of the possible angular velocities. Once R_2 reaches E, it approximates EE' with a line and continues reducing the lateral error.

VIII. Conclusion

In this paper, we presented a decentralized near-to-near algorithm for platooning along a path generated arbitrary as the leader moves. Instead of tracking successive positions, each following robot realizes an on-line generation of its predecessor's path. Then, it considers a sequence of angular velocities and selects the one for which the lateral deviation between the follower and its predecessor's path is the smallest. The selection in this sequence is realized using an heuristic search. Numerical experiments on a seven-robot convoy showed that the accumulated lateral error along the convoy remains small. However, this error does not vanish due to the path approximation and the discretization of the command domain. Moreover, this lateral error may also result from the non-fulfillment of the initialization condition when the target path changes. Also steering with the maximum angular velocity to avoid crossing the leader's path generates a peak of lateral error which is quickly reduced by steering with the maximum angular velocity in the opposite direction. In the future works, we will try to improve the path approximation and find a more intelligent way for discretizing the angular velocity domain. Also, in order to reduce the lateral deviation due to the reasons given above, we propose in the future works, to improve our optimization criteria by planning the follower's actions over several time steps. Given that the follower's longitudinal velocity influences the angular velocity computation, the effects of the longitudinal control on the lateral behavior will also be studied. Moreover, numerical simulations were done in a perfect world hypothesis where neither delays nor noise induced by the sensors were considered. In the future works, we will implement the proposed approach on real robots in order to study the robustness to measurement noise and model uncertainties.

Acknowledgement

This work was partially supported by INTRADE project.

References

[1] G. Antonelli and S. Chiaverini, "Kinematic control of platoons of autonomous vehicles," *IEEE Trans. Robotics*, vol. 22, no. 6, pp. 1285–1292, Dec. 2006.

[2] C. Smaili, M. El Badaoui El Najjar, and F. Charpillet, "Multi-sensor fusion method using bayesian network for precise multi-vehicle localization," in *11th International IEEE Conference on Intelligent Transportation Systems*, Beijing (CN), Oct. 2008, pp. 906—-911.

[3] J. Bom, B. Thuilot, F. Marmoiton, and P. Martinet, "Nonlinear control for urban vehicles platooning, relying upon a unique kinematic gps," in *International Conference on Robotics and Automation*, Barcelona, Spain, 2005, pp. 41–46.

[4] P. Avanzini, B. Thuilot, T. Dallej, P. Martinet, and J. Derutin, "On-line reference trajectory generation for manually convoying a platoon of automatic urban vehicles," in *IROS*, 2009, pp. 1867–1872.

[5] P. Avanzini, B. Thuilot, and P. Martinet, "Obstacle avoidance and trajectory replanification for a group of communicating vehicles," in *ITST*, 2009, pp. 267 – 272.

[6] P. Kavathekar and Y. Chen, "Draft: Vehicle platooning: a brief survey and categorization," in *Proceedings of The ASME 2011 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2011.

[7] J.-M. Contet, F. Gechter, P. Gruer, and A. Koukam, "Multiagent system model for vehicle platooning with merge and split capabilities," in *3rd International Conference on Autonomous Robots and Agents (ICARA)*, Palmerston North (NZ), Dec. 2006.

[8] S.-Y. Yi and K.-T. Chong, "Impedance control for a vehicle platoon system," *Mechatronics*, vol. 15, no. 5, pp. 627–638, June 2005.

[9] P. Daviet and M. Parent, "Longitudinal and lateral servoing of vehicles in a platoon," in *Proc. of the IEEE Int. Symp. on Intelligent Vehicles*, Tokyo (JP), Sept. 1996, pp. 41–46.

[10] J. Yazbeck, A. Scheuer, O. Simonin, and F. Charpillet, "Improving near-to-near lateral control of platoons without communication," in *IROS*, 2011, pp. 4103–4108.

[11] A. Scheuer, O. Simonin, and F. Charpillet, "Safe longitudinal platoons of vehicles without communication," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, Kobe (JP), May 2009, pp. 70–75, http://www.loria.fr/~scheuer/Platoon.