Cooperative Nonlinear Model Predictive Control for Flocks of Vehicles

Y. Rochefort [*] **S. Bertrand** [*] **H. Piet-Lahanier** [*] **D. Beauvois** [**]
D. Dumur [**]

[*] *ONERA - The French Aerospace Lab, Palaiseau, France (e-mail: yohan.rochefort, helene.piet-lahanier, sylvain.bertrand at onera.fr).*
[**] *SUPELEC Systems Sciences E3S, Gif-Sur-Yvette, France (e-mail: dominique.beauvois, didier.dumur at supelec.fr).*

Abstract: This paper describes the guidance of a group of autonomous cooperating vehicles using model predictive control. The developed control strategy allows to find a feasible near optimal control sequence with a short and constant computation delay in all situations. It makes use of the nonlinear model of the vehicle and takes other vehicle intentions into account. Numerical simulations are provided where a group of vehicles must reach several way-points while avoiding obstacles and collisions inside the group. These simulations allow to compare computation delay and efficiency of the proposed approach with traditional optimisation.

Keywords: Cooperative Nonlinear Model predictive control; Distributed control; Multi-vehicle systems; Autonomous mobile robots; Obstacle avoidance; Real time control

1. INTRODUCTION

Cooperative control stems from the idea that a group of several (possibly heterogeneous) cooperating vehicles might be more efficient and robust in the completion of complex tasks than a single vehicle. Cooperation can be achieved with centralized (Wang et al. (2007)) or distributed control (Rochefort et al. (2011), Siva and Maciejowski (2011)). In a centralized scheme, actions of all vehicles are easily synchronized. Distributed control on the other hand allows better scalability: each vehicle carries the burden of its own control input computation. Our approach is to distribute the computation of the control input and ensure cooperation by communicating intentions and newly acquired information among vehicles.

When designing a cooperative control law, it is natural to look toward Model Predictive Control (MPC). The main advantage of MPC is to anticipate future consequences of a control input. This is specifically interesting to take the intentions of other vehicles into account. Moreover, MPC allows to deal with fulfilment of different objectives in complex missions by means of cost functions.

MPC is defined by on-the-fly resolution of an open-loop optimal control problem at regular time intervals during the mission. This optimal problem consists in finding the control sequence which minimizes a cost function over a horizon of prediction. Only the first part of this optimal control sequence is applied until the next control problem is solved with updated information. This allows the vehicle to react to unforeseen events.

Main drawback of MPC is the unpredictable computation delay of the optimisation procedure. Current research on cooperative distributed MPC focuses mainly on proof of convergence (Dunbar and Murray (2004), Müller et al. (2011)), robustness (Siva and Maciejowski (2011)), and formation flying (Olfati-Saber et al. (2003)) without detailing the computation delay. In the meantime, real time feasibility of Nonlinear MPC (NMPC) is studied for fast dynamic systems, like the Caltech ducted fan (Dunbar (2001)) or a twin-pendulum (Alamir and Murilo (2008)).

This paper presents a NMPC based distributed algorithm that computes a near optimal feasible control sequence for each member of a group of cooperating vehicles. The group of vehicles must travel in an unknown environment to sequentially reach several way-points. Meanwhile it must avoid collisions (within the group and with other obstacles) and remain gathered. The designed algorithm runs with a constant computation delay in all circumstances and faster than an algorithm using traditional optimisation.

The proposed approach is inspired by Frew (2006). In this work Frew used a random search to find a near optimal feasible control sequence for one vehicle in an unknown environment with obstacles. In our work, a parsimonious systematic search of the command space is used instead of a random search. The objective function is also greatly modified to extend the technique to multi-vehicle systems.

Section 2 states the problem. Section 3 contains a short explanation of the MPC approach whereas the objective function is detailed in section 4. Section 5 describes the control sequence selection. Numerical simulations are provided in section 6 to illustrate and analyse the performances of the proposed approach. Conclusion and perspectives of works end this paper.

2. PROBLEM STATEMENT

A system composed of N autonomous vehicles moving on a horizontal plane is considered. This allows representation of a group of ground vehicles or a flock of aerial vehicles maintaining a constant altitude. The dynamics of a given vehicle i of the group are assumed to be represented by the following discrete-time kinematic model:

$$x_i(k+1) = x_i(k) + \Delta t.v_i(k).\cos(\Psi_i(k)) \quad (1)$$
$$y_i(k+1) = y_i(k) + \Delta t.v_i(k).\sin(\Psi_i(k)) \quad (2)$$
$$v_i(k+1) = v_i(k) + \Delta v_i(k) \quad (3)$$
$$\Psi_i(k+1) = \Psi_i(k) + \Delta t.\omega_i(k) \quad (4)$$
$$\omega_i(k+1) = \omega_i(k) + \Delta\omega_i(k) \quad (5)$$

where $p_i(k) = [x_i(k)\ y_i(k)]^T$ is the position of vehicle i in $\mathcal{F}$ at step k; $\mathcal{F}$ is the common inertial frame used as a reference to communicate informations between vehicles; Δt is the sampling time; $v_i(k)$ is the linear speed of vehicle i in $\mathcal{F}$ at step k (linear will be omitted from now on); $\Psi_i(k)$ is the orientation angle of vehicle i in $\mathcal{F}$ at step k; $\omega_i(k)$ is the turn rate of vehicle i at step k. The control input of vehicle i at step k consists of a speed increment and a turn rate increment: $u_i(k) = [\Delta v_i(k)\ \Delta\omega_i(k)]^T$.

Kinematics of vehicles is constrained by practical limitations which are identical for all vehicles. These limitations apply at every step k on the speed, on the turn rate and on the control input of the vehicles.

$$v_{min} \leqslant v_i(k) \leqslant v_{max} \quad (6)$$
$$-\omega_{max} \leqslant \omega_i(k) \leqslant \omega_{max} \quad (7)$$
$$-\Delta v_{max} \leqslant \Delta v_i(k) \leqslant \Delta v_{max} \quad (8)$$
$$-\Delta\omega_{max} \leqslant \Delta\omega_i(k) \leqslant \Delta\omega_{max} \quad (9)$$

Our objective is to guide the vehicles in the environment to reach several way-points. Meanwhile, vehicles must avoid collisions with each other and external obstacles. When possible, the vehicles must travel as a flock and at nominal speed v_n. We consider that the vehicles form a flock when the distance between nearest neighbours is approximately d_{des} and no vehicle is at a distance greater than $d_{ign} > d_{des}$ from the remaining of the flock. d_{ign} represents the loss of communication and sensing with too distant vehicles.

To ensure collision and obstacle avoidance the distance between a vehicle and a threat must be greater than a threshold $d_{saf} < d_{des}$. This threshold is different for vehicle to vehicle and vehicle to obstacle avoidances.

To preserve scalability and ensure robustness to single vehicle failure, each vehicle must compute its own control input. Cooperation takes the form of information sharing between vehicles. The main information shared is the path that the vehicles intend to follow in the future, described in the common inertial frame $\mathcal{F}$. Any other information sensed by a vehicle (*e.g.* position of external obstacle) is also transmitted to the rest of the group.

3. MODEL PREDICTIVE CONTROL (MPC)

Each vehicle computes its control input using a MPC approach. It consists of repeatedly solving an open-loop optimal control problem at regular time intervals. At step k, the open-loop problem for vehicle i is defined by:

Find the control sequence of length H_c (control horizon) $\mathbf{u_i^*}(k+1) = [u_i^*(k+1) \cdots u_i^*(k+H_c)]$ that minimizes an objective function J.

The objective function J is evaluated over the time interval of length H_p (prediction horizon) by predicting the effects of the control sequence using the model of the vehicles. Note that $2 \leqslant H_c \leqslant H_p \leqslant \infty$ and that the applied control is null after H_c.

The effectiveness of the model predictive control strategy is largely determined by the objective function minimized in the problem above. Assuming this problem can be solved in a time lower than Δt, the model predictive approach consists of two phases, repeated until the goal is reached:

(1) At step k, each vehicle i computes its optimal control sequence $\mathbf{u_i^*}(k+1)$
(2) At step $k+1$, each vehicle applies the first term of this control sequence.

4. OBJECTIVE FUNCTION DEFINITION

To compute the cost J_i^{rhc} associated with one control sequence $\mathbf{u_i(k+1)}$, each vehicle i predicts the evolution of its position, speed and turn rate caused by the application of the entire control sequence according to the nonlinear model of the vehicle (equ. 1 to 5). The only information available on the other cooperating vehicles is the trajectory prediction transmitted at the previous step.

Predicted values of parameters (the vehicle own parameters and other vehicle predictions) are distinguished from the real values by adding a hat on the notations.

4.1 Definition of costs

The objective function J_i^{rhc} that is to be minimized by the vehicle i consists of three main costs: control cost J_i^u, mission cost J_i^m, and cluster cost J_i^c. These costs include several components, each one handling one aspect of the mission.

$$J_i^{rhc}(k) = J_i^u(k) + J_i^m(k) + J_i^c(k) \quad (10)$$

Control cost J_i^u – This cost aims at moderating the amplitude of the control inputs and therefore the energy used to control the vehicle. It includes the cost of speed increment $J_i^{\Delta v}$ and the cost of turn rate increment $J_i^{\Delta\omega}$.

$$J_i^u(k) = J_i^{\Delta v}(k) + J_i^{\Delta\omega}(k) \quad (11)$$

$$J_i^{\Delta v}(k) = W^{\Delta v}.\sum_{n=k+1}^{k+H_c} \Delta v_i^2(n) \quad (12)$$

$$J_i^{\Delta\omega}(k) = W^{\Delta\omega}.\sum_{n=k+1}^{k+H_c} \Delta\omega_i^2(n) \quad (13)$$

The weighting coefficients $W^\bullet$ will be explained later.

Mission cost J_i^m – This cost aims at ensuring mission completion. It includes four components; J_i^{mv} which purpose is to travel at nominal speed; $J_i^{m\omega}$ which purpose is to favour straight lines; J_i^{mt} which purpose is to make the vehicle move along a straight line reference trajectory toward the next way-point; and J_i^{mf} which purpose is to make the vehicle move closer to the way-point. Last two costs gain no benefit if nominal speed is exceeded to move closer to the way-point.

$$J_i^m(k) = J_i^{mv}(k) + J_i^{m\omega}(k) + J_i^{mf}(k) + J_i^{mt}(k) \quad (14)$$

$$J_i^{mv}(k) = W^{mv}.\sum_{n=k+1}^{k+H_p} \left(\widehat{v}_i(n) - v_n\right)^2 \quad (15)$$

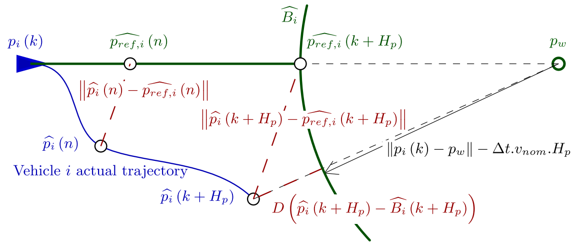

$\widehat{B}_i$
$p_i(k)$
$\widehat{p_{ref,i}}(n)$
$\widehat{p_{ref,i}}(k+H_p)$
p_w
$\left\|\widehat{p}_i(n) - \widehat{p_{ref,i}}(n)\right\|$
$\left\|\widehat{p}_i(k+H_p) - \widehat{p_{ref,i}}(k+H_p)\right\|$
$\widehat{p}_i(n)$
$\left\|p_i(k) - p_w\right\| - \Delta t.v_{nom}.H_p$
Vehicle i actual trajectory
$\widehat{p}_i(k+H_p)$
$D\left(\widehat{p}_i(k+H_p) - \widehat{B}_i(k+H_p)\right)$

Fig. 1. Representation of $\widehat{B}$ and $\widehat{p_{ref,i}}$

$$J_i^{m\omega}(k) = W^{m\omega}.\sum_{n=k+1}^{k+H_p} \widehat{\omega}_i(n)^2 \tag{16}$$

$$J_i^{mt}(k) = W^{mt}.\sum_{n=k+1}^{k+H_p} \left\|\widehat{p}_i(n) - \widehat{p_{ref,i}}(n)\right\|^2 \tag{17}$$

$$J_i^{mf}(k) = W^{mf}.\left(D\left(\widehat{p}_i(k+H_p), \widehat{B}_i(k+H_p)\right)\right)^2 \tag{18}$$

At all times, only the currently aimed way-point is considered. This way-point changes for all vehicles as soon as the distance between one of them and the way-point is lower than $\Delta t.v_n.H_p$.

$\widehat{p_{ref,i}}(n)$ is a reference point defined as the position reached at step n by the vehicle i if it moves straight toward the way-point position p_w at nominal speed. The reference points are given by the equation bellow.

$$\widehat{p_{ref,i}}(n) = p_i(k) + (n-k).\Delta t.v_n.\frac{p_i(k) - p_w}{\|p_i(k) - p_w\|} \tag{19}$$

$\widehat{B}_i(k+H_p)$ is the smallest ball surrounding the way-point that contains $\widehat{p_{ref,i}}(k+H_p)$. *i.e.* $\widehat{B}_i(k+H_p)$ is the smallest ball around the way-point reachable at nominal speed in an ideal scenario where the vehicle moves directly toward the way-point. $D(\widehat{p}_i(k+H_p), \widehat{B}_i(k+H_p))$ represents the distance between the predicted position of vehicle i and ball $\widehat{B}_i$.

These formulations ensure the shape and dynamic of the costs J_i^{mt} and J_i^{mf} remain the same at all times during the mission whatever the distance to the way-point. Figure 1 depicts the reference points and ball $\widehat{B}_i$.

J_i^{mf} and J_i^{mt} are complementary. J_i^{mf} will ensure the vehicle moves closer to the way-point even if straight line is not possible. J_i^{mt} will influence the vehicle to move directly toward the way-point instead of turning around while approaching (*i.e.* spiralling) toward the way-point.

Cluster cost J_i^c – This cost aims at ensuring collision avoidance and keeping all vehicles grouped together. It includes three components; J_i^{ca} which ensures collision avoidance between cooperating vehicles; J_i^{co} which ensures collision avoidance with external obstacles; and J_i^{cf} which is responsible for maintaining the vehicles close to each other.

$$J_i^c(k) = J_i^{ca}(k) + J_i^{cf}(k) + J_i^{co}(k) \tag{20}$$

$$J_i^{ca}(k) = W^{ca}.\sum_{j=1}^{N}\sum_{n=k+1}^{k+H_p} \frac{1 - tanh\left(\left(\widehat{d}_{ij}(n) - \beta_a\right).\alpha_a\right)}{2} \tag{21}$$

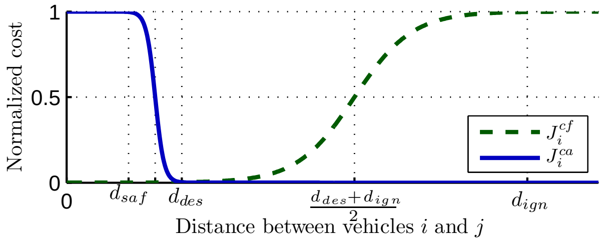


Fig. 2. Shape of avoidance and flocking cost

$$J_i^{co}(k) = W^{co}.\sum_{j=1}^{N_o}\sum_{n=k+1}^{k+H_p} \frac{1 - tanh\left(\left(\widehat{d}_{ij}(n) - \beta_a\right).\alpha_a\right)}{2} \tag{22}$$

$$J_i^{cf}(k) = W^{cf}.\sum_{j=1}^{N}\sum_{n=k+1}^{k+H_p} \frac{1 + tanh\left(\left(\widehat{d}_{ij}(n) - \beta_f\right).\alpha_f\right)}{2} \tag{23}$$

N_o is the number of obstacles; $\widehat{d}_{ij}(n)$ is the distance separating the vehicle i from the threat j (an obstacle or another vehicle) at step n. The choice of hyperbolic tangent for this cost is driven by two reasons. First the shape of the function: a slope between two nearly flat regions with quick transitions and differentiable everywhere as shown in figure 2. Second the function is fast to compute (*e.g.* fastest than the error function $erf(d) = \frac{2}{\sqrt{\pi}}\int_0^d e^{-t^2}dt$).

The terms $\alpha_\bullet$ and $\beta_\bullet$ are used to shape the hyperbolic tangent to our needs. To explain this, let us take the example of the vehicle avoidance cost J_i^{ca}. The term α_a allows to define the width of the region where the function varies rapidly. The term β_a allows to set the inflexion point of the function in the abscissa axis. It is desirable that the cost be nearly constant outside of $[d_{saf}\ d_{des}]$ and vary rapidly inside this region. Thus the terms are tuned to ensure the derivative of the cost is lower than 0.05 outside $[d_{saf}\ d_{des}]$ and greater inside. They depend on the values of the characteristic distances d_{saf}, d_{des}, and d_{ign}.

$$\alpha_a = 6.(d_{des} - d_{saf})^{-1} \tag{24}$$

$$\beta_a = {}^1\!/\!_2.(d_{des} + d_{saf}) \tag{25}$$

$$\alpha_f = 6.(d_{ign} - d_{des})^{-1} \tag{26}$$

$$\beta_f = {}^1\!/\!_2.(d_{ign} + d_{des}) \tag{27}$$

4.2 Definition of weighting coefficients

Each component of the objective function is weighted according to its relative priority, that is the importance of the corresponding task in the mission. As an example, remaining a group could be more important than travelling at nominal speed but less important than collision avoidance. Therefore, the group may split to avoid collision, but otherwise vehicles would adapt their speed to stay together.

The weights $W^\bullet = w^\bullet.k^\bullet$ consist of:

$k^\bullet$ a normalisation coefficient, used to gather all components of the objective function in a same order of magnitude to ease the tuning of the weighting factors;

$w^\bullet$ a tuning parameter used to control the relative importance of the components of the objective function.

Table 1. Normalization coefficients

$k^{\Delta v} = \left(H_c.\Delta v_{max}^2\right)^{-1}$	$k^{mf} = \left((H_p.\Delta t.v_n)^2\right)^{-1}$
$k^{\Delta \omega} = \left(H_c.\Delta \omega_{max}^2\right)^{-1}$	$k^{ca} = (H_p/2)^{-1}$
$k^{m\omega} = \left(H_c.\omega_{max}^2\right)^{-1}$	$k^{co} = (H_p/2)^{-1}$
$k^{mt} = \left(\sum_{n=1}^{H_p} (n.\Delta t.v_n)^2\right)^{-1}$	$k^{cf} = (H_p.N)^{-1}$
$k^{mv} = \left(H_c.max\left(v_n - v_{min}, v_{max} - v_n\right)^2\right)^{-1}$	

To define the normalization coefficients, we use reference scenarios whose normalized cost must be one. For most of the components, we define this scenario as the worst movement of the vehicle that does not compromise the mission (*e.g.* highest turn rate allowed for the entire control horizon for $k^{\Delta\omega}$). One exception is the progression toward the way-point (J_i^{mt} and J_i^{mf}), which uses a scenario where the vehicle does not move. Second exception is the *travelling as a group* component which uses a scenario where the vehicle is at a distance greater than d_{ign} from all the others.

Normalization coefficients are given in table 1.

5. CONTROL SEQUENCE SEARCH PROCEDURE

To simplify our optimisation problem (and decrease computation delay), a constant control input is applied during the entire control horizon or until the maximal value for speed or turn-rate is reached. A null control sequence is then applied for the remaining of the prediction horizon. Because short control sequences are used and because control inputs are variations of speed and turn rate, this simplification does not impair vehicle capabilities. The optimisation procedure must thus find only two elements: a speed increment Δv and a turn rate increment $\Delta\omega$.

To avoid the unpredictability of the delay introduced by the optimisation procedure, a search procedure among a finite set of admissible control sequences have been chosen. Doing so, the amount of computation remains the same in all circumstances. This procedure has four phases.

(1) Define a finite set of candidate control sequences $\mathcal{C}_u = \mathcal{C}_{\Delta v} \times \mathcal{C}_{\Delta\omega}$. Each candidate consists of a sequence of speed increment and a sequence of turn rate increment. It must respect the constraints on control input, speed, and turn rate. The appropriate number of candidates is discussed in section 5.2;
(2) Using the nonlinear model (equ 1 to 5), predict the trajectory that corresponds to each candidate;
(3) Compute the cost of each candidate;
(4) Use the candidate that implies the smallest cost.

Note that the time needed to find the smallest value among a discrete set of scalar values is negligible compared to the time necessary to predict the trajectories.

5.1 Distribution of the candidates

The subtlety in our method lies in the creation of the set of candidates, and more precisely, in their distribution in the control space. At first glance, three possible ways are:

(1) generate many control sequences distributed uniformly over the control space. This will allow precise control, but will take time to predict all trajectories;
(2) generate a moderate amount of uniformly distributed control sequences. This will be fast but the control may lack precision (oscillations, missed possibilities)
(3) generate a moderate amount of control sequences randomly distributed over the control space. This is the approach of Frew (2006) (but with constant speed). It is fast and as the distribution changes iteration after iteration, the probability to find the control that will make the oscillation cease or discover a new solution increases. A set of predefined control sequences must be added to the randomly generated set to ensure that particular trajectories are always possible (like straight line, or maximum turn rate).

In this work another approach has been chosen, which is to use a small amount of control sequences distributed in particular way over the control space. Our distribution intends to implement the following intuition :

At the beginning of a high amplitude manoeuvre (like a u-turn or an emergency brake), a precise control input is not necessary because the amplitude is the main concern. On the other hand, as manoeuvre comes to an end or for small amplitude manoeuvre, high precision is desirable.

This intuition motivates the three following rules.

(1) The set of candidates includes the extreme control inputs (that is $\pm\Delta v_{max}$, and $\pm\Delta\omega_{max}$) to exploit the full potential of the vehicle;
(2) The set of candidates includes the null control input (that is $\Delta v = 0$, and $\Delta\omega = 0$) to allow to continue with the same speed and turn rate;
(3) Candidates are distributed over the entire control space with an increased density around zero.

To meet these rules, we have chosen to use an inverse power function and add the null control input.

$$\mathcal{C}_{\Delta v} = \left\{ \pm \frac{\Delta v_{max}}{\varphi^p} \right\} \cup \{0\} \qquad \text{with } p = 0 \text{ to } \eta_{\Delta v} \qquad (28)$$

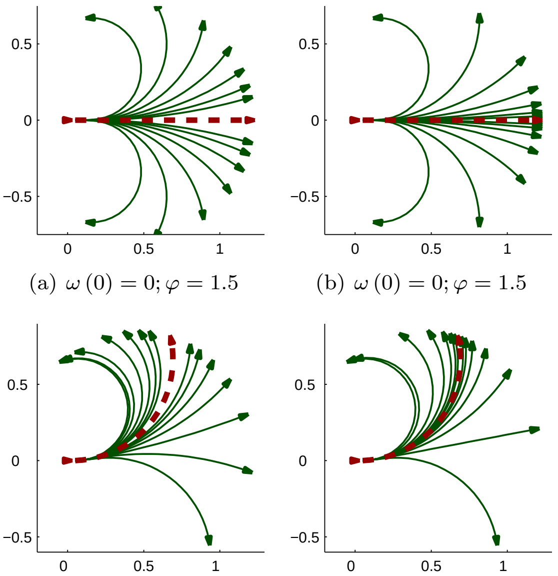
$$\mathcal{C}_{\Delta \omega} = \left\{ \pm \frac{\Delta \omega_{max}}{\varphi^p} \right\} \cup \{0\} \qquad \text{with } p = 0 \text{ to } \eta_{\Delta \omega} \qquad (29)$$

$\mathcal{C}_{\Delta v}$ is the set of candidates of speed increments; $\mathcal{C}_{\Delta\omega}$ is the set of candidates of turn rate increments; φ controls the interval between two candidates, its value will be studied later; $\eta_{\Delta v}$ and $\eta_{\Delta\omega}$ define the smallest available control input ($\neq 0$), which is also the highest possible precision of our control strategy. Figure 3 shows the trajectories generated by the different values in $\mathcal{C}_{\Delta\omega}$ for two different initial turn rates and two values of φ. In this figure, the speed of the vehicle is kept constant at v_n.

5.2 Discussion on the number of candidates

The particular distribution that has been chosen is an attempt to explore the control space efficiently, *i.e.* explore all of it but insist on the most useful part to reduce the amount of computation. The definition of these distributions links the maximum possible precision of the control input $\eta_{\Delta v}$ and $\eta_{\Delta\omega}$ to the number of candidates $N_{\Delta v}$ and $N_{\Delta\omega}$ with $N_\bullet = 2 * \eta_\bullet + 1$.

To explore systematically the control space, each candidate of speed increment will be combined with each candidate of turn rate increment to form $N_{\Delta v}.N_{\Delta\omega}$ can-



(a) $\omega\left(0\right)=0;\varphi=1.5$

(b) $\omega\left(0\right)=0;\varphi=1.5$

(c) $\omega\left(0\right)=\frac{\omega_{max}}{2};\varphi=2$

(d) $\omega\left(0\right)=\frac{\omega_{max}}{2};\varphi=2$

Fig. 3. Trajectories predicted from the set of candidates of turn rate increments

Table 2. Simulation parameters

v_n	$0.1\ m.s^{-1}$	H_c	$4\ \Delta t$
v_{max}	$0.2\ m.s^{-1}$	H_p	$24\ \Delta t$
v_{min}	$0.05\ m.s^{-1}$	d_{saf}	$0.7\ m$
Δv_{max}	$0.02\ m.s^{-2}$	d_{des}	$1.3\ m$
ω_{max}	$0.3\ rad.s^{-1}$	d_{ign}	$5\ m$
$\Delta\omega_{max}$	$0.15\ rad.s^{-2}$	$N_{\Delta v}$	5
Δt	$0.5\ s$	$N_{\Delta\omega}$	15

didate control sequences. As each candidate will be used to predict a trajectory and evaluate the associated cost, the amount of computation is directly proportional to the number of candidates.

The values of $\eta_{\Delta v}$ and $\eta_{\Delta\omega}$ must be chosen while considering different things: the precision of the actuators: it is unnecessary to test two control inputs that will be executed in the same way; the precision of the available measurements: it is unnecessary to test two control inputs which executions will be undistinguishable; the importance of precise guidance and computation capacity of the on-board computer: more precision means more computation delay.

6. NUMERICAL SIMULATION

The proposed search scheme is tested on the scenario presented in figure 4. Initial positions and orientations of the vehicles are randomly chosen in the region $-12.5; m \leqslant x \leqslant -7.5; m; -3.5; m \leqslant y \leqslant 1.5; m; -\pi \leqslant \theta \leqslant \pi$. Vehicles must reach the various way-points in the indicated order. Mission is a success if the group of vehicles reaches the third way-point within $500s$. Mission fails if one vehicle is separated from the flock by more than d_{ign} or if a collision arises.

Figure 4 depicts an example of a simulation done with the proposed search scheme using the parameters and weights defined in table 2 and 3. Unless stated otherwise, these parameters were used in all simulations.

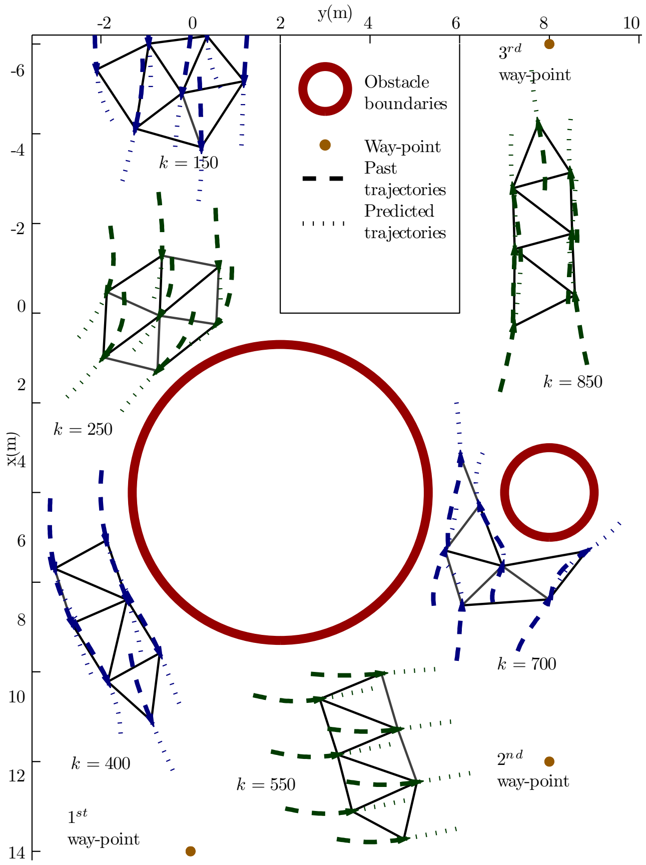


Fig. 4. Test scenario and example of realisation

Table 3. Weighting coefficients

$w_i^{\Delta v}$	2	$w_i^{m\omega}$	5	w_i^{ca}	100
$w_i^{\Delta\omega}$	10	w_i^{mf}	10	w_i^{cf}	50
w_i^{mv}	5	w_i^{mt}	5	w_i^{co}	200

Our test procedure consisted in solving repeatedly (500 times) this scenario while randomly changing the initial positions and orientations of the vehicles. Each column of table 4 contains the results obtained with a specific set of parameters. Using the set given in table 2 as a base, the influence of three parameters specific to our scheme of search was studied: the number of candidate speed increments $N_{\Delta v}$; the number of candidate turn rate increments $N_{\Delta\omega}$; and the repartition of the candidates defined by φ. Last column contains the results obtained using a traditional optimisation (Matlab *fmincon*) instead.

Failure and success rates for each set of parameters are given in table 4 (Collisions, Lost vehicles, Success). It contains also the average of the following values, computed on the successful missions only: time needed to complete the mission $\overline{t_{arrival}}$; computation delay (to compute the control input of one vehicle, $\overline{t_{computation}}$); total cost of the mission detailed in three parts : control cost $\overline{J_u}$, mission cost $\overline{J_m}$ and cluster cost $\overline{J_c}$.

6.1 Comparison with a traditional optimisation procedure

The optimisation algorithm we have chosen to make this comparison is *fmincon*, because it is readily available in Matlab. Our objective, as already stated, is to find a control scheme that has a constant and short computa-

Table 4. Success and average efficiency over 500 simulations for several sets of parameters

$N_{\Delta v}/N_{\Delta \omega}/\varphi$	5/15/1.75	3/15/1.75	7/15/1.75	5/11/1.75	5/19/1.75	5/15/1.5	5/15/2	*fmincon*
Collisions	10	11	27	16	14	10	11	15
Lost vehicles	21	36	41	20	41	18	38	69
Success	469	453	432	464	445	472	451	416
$\overline{t_{arrival}}$	$437s$	$415s$	$443s$	$438s$	$436s$	$437s$	$436s$	$450s$
$\overline{t_{computation}}$	$11.4ms$	$7.9ms$	$15.6ms$	$9.5ms$	$14.3ms$	$12.0ms$	$12.0ms$	$85.2ms$
$\overline{J_u}$	315	317	291	346	309	343	306	310
$\overline{J_m}$	7152	8495	7033	7248	7210	7202	7221	6860
$\overline{J_c}$	53543	84631	45970	54591	55314	54432	54380	43326

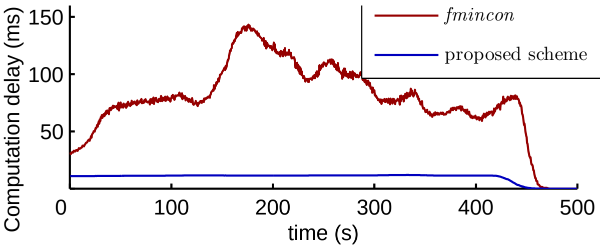


Fig. 5. Computation delay during the mission

tion delay with performance comparable to a traditional optimisation.

The simulations done with our search scheme present a computation delay five to seven time shorter than *fmincon*. In addition, figure 5 pictures the average computation delay during the mission. While it stays constant with our scheme, the computation delay induced by *fmincon* varies greatly depending on the situation. In particular, the optimisation takes two times longer if an obstacle is present. In the same time, our search scheme is a little more efficient at avoiding lost vehicle, but leads to slightly higher costs.

6.2 Influence of the number of candidates

As stated in section 5.2, the control becomes more precise when the number of candidates increases. This claim is supported by the observation of the cluster cost for the various values of $N_{\Delta v}$: with a more precise control of the speed, the vehicles can stay closer to each other and the associated cost decreases. As this observation is not true for $N_{\Delta v}$, it is possible that increasing the precision of control has little influence beyond a threshold.

A very straightforward observation is the increase of computation delay $\overline{t_{computation}}$ when the number of candidate control sequences increases.

6.3 Influence of candidate distributions

As stated in 5.1, the particular distribution of the candidates allows to insist on the more useful parts while exploring the entire control space. The question arises of the distribution that will produce the best performance. As it causes more lost vehicles, the value $\varphi = 2$ can be excluded. But $\varphi = 1.5$ and $\varphi = 1.75$ can not be distinguished one from the other.

7. CONCLUSION

In this paper, an algorithm has been proposed for the distributed control of a group of cooperative vehicles. This algorithm consists in the resolution of a NMPC problem using an efficient search approach instead of classical optimization. The advantage of this scheme is that the induced computation delay is shorter and constant in all situations without penalizing efficiency. These properties make this algorithm ideal for embedded control. Numerical simulations have been produced to support these claims.

The influence of the number of candidates and their repartition over the control space have been studied. Several sets of parameters offer comparable efficiencies. Additional work will be done to distinguish between the various possibilities.

Future work will focus on the objective function and on the relative weights. To avoid collisions, a system of priority could be implemented among vehicles.

REFERENCES

Alamir, M. and Murilo, A. (2008). Swing-up and stabilization of a twin-pendulum under state and control constraints by a fast nmpc scheme. *Automatica*, 44(5), 1319–1324.

Dunbar, W. and Murray, R. (2004). Receding horizon control of multi-vehicle formations: a distributed implementation. In *Proceedings of the 43rd Conference on Decision and Control*, volume 2, 1995 – 2002.

Dunbar, W.B. (2001). Model predictive control: Extension to coordinated multi-vehicle formations and real-time implementation. PHD Candidacy report, Control and Dynamical Systems, California Institute of Technology.

Frew, E.W. (2006). Receding horizon control using random search for uav navigation with passive, non-cooperative sensing. Technical report, University of Colorado at Boulder.

Müller, M.A., Reble, M., and Allgöwer, F. (2011). A general distributed mpc framework for cooperative control. In *Proceedings of the 18th IFAC World Congress.*

Olfati-Saber, R., Dunbar, W., and Murray, R. (2003). Cooperative control of multi-vehicle systems using cost graphs and optimization. In *Proceedings of the American Control Conference*, volume 3, 2217 – 2222.

Rochefort, Y., Piet-Lahanier, H., Bertrand, S., Beauvois, D., and Dumur, D. (2011). Guidance of flocks of vehicles using virtual signposts. In *Proceedings of the 18th IFAC World Congress.*

Siva, E. and Maciejowski, J.M. (2011). Robust multiplexed mpc for distributed multi-agent systems. In *Proceedings of the 18th IFAC World Congress.*

Wang, X., Yadav, V., and Balakrishnan, S.N. (2007). Cooperative uav formation flying with obstacle/collision avoidance. *IEEE Transactions on Control Systems Technology*, 15(4), 672–679.