A DECENTRALIZED COLLISION AVOIDANCE ALGORITHM FOR MULTI-ROBOTS NAVIGATION

Michael Defoort, Arnaud Doniec and Noury Bouraqadi
Ecole de Mines de Douai, 941 rue Charles Bourseul, BP 10838, 59508 Douai, France

Keywords: Decentralized intelligence, Real-time path planning, Collision avoidance, Receding horizon, Nonholonomic mobile robots.

Abstract: This paper presents a distributed strategy for the navigation of multiple autonomous robots. The proposed planning scheme combines a decentralized receding horizon motion planner, in which each robot computes its own planned trajectory locally, with a fast navigation controller based on artificial potential fields and sliding mode control technique. This algorithm solves the collision avoidance problem. It explicitly accounts for computation time and is decentralized, making it suited for real time applications. Simulation studies are provided in order to show the effectiveness of the proposed approach.

1 INTRODUCTION

The research effort in multi-robots systems (MRS) relies on the fact that multiple robots have the possibility to perform a mission more efficiently than a single robot. Among all the topics of study in this field, the issue of conflict resolution becomes an increasingly important point. Many cooperative tasks such as surveillance, search, rescue or area data acquisition need the robots to autonomously navigate without collision.

Solving conflicts in MRS consists in introducing some coordination mechanisms in order to give a coherence between the robot acts (Kuchar and Yang, 2000). For motion planning, three coordination mechanisms are identified:

- *The Coordination by Adjustment*, where each robot adapts its behavior to achieve a common objective (Tomlin et al., 1998). However, most of the planning algorithms are centralized, which often limit their applicability in real systems.
- *The Coordination by Leadership (or supervision)* where a hierarchical relationship exists between robots (Das et al., 2002). Such an approach is easy to implement. However, due to the lack of an explicit feedback from the followers to the leader, the collision avoidance cannot be guaranteed if followers are perturbed (during obstacle avoidance for instance). Another disadvantage is that the leader is a single point of failure.
- *The Standardization* where procedures are predefined to solve some particular interaction cases (Pallatino et al., 2007). While this approach can lead to straightforward proofs, it also tends to be less flexible with respect to changing conditions.

Here, the problem of interest is the decentralized navigation for autonomous robots through a coordination by adjustment. Each vehicle is modeled as an unicycle with a limited sensing range in order to capture the essential properties of a wide range of vehicles. They are dynamically decoupled but have common constraints that make some conflicts. Indeed, each robot has to avoid collision with the other entities. Furthermore, the proposed framework allows moving (and static) obstacles to be avoided since they can be modeled as non cooperative entities.

Motion planning consists in generating a collision-free trajectory from the initial to the final desired positions for a robot. Since the environment is partially known and further explored in real time, the computation of complete trajectories from start until finish must be avoided. Therefore, the trajectories have to be computed gradually over time while the mission unfolds. It can be accomplished using an online receding horizon planner (Mayne et al., 2000), in which partial trajectories from an initial state toward the goal are computed by solving optimal control problems over a limited horizon.

Two strategies for motion planning in MRS

Michael D., Doniec A. and Bouraqadi N. (2009).
A DECENTRALIZED COLLISION AVOIDANCE ALGORITHM FOR MULTI-ROBOTS NAVIGATION.
In *Proceedings of the 6th International Conference on Informatics in Control, Automation and Robotics - Robotics and Automation*, pages 44-51
DOI: 10.5220/0002194900440051

are the centralized and decentralized (distributed) approaches. Although the centralized one has been used in different studies (see (Dunbar and Murray, 2002) for instance), its computation time which scales exponentially with the number of robots, its communication requirement and its lack of security make it prohibitive. To overcome these limitations, one can use a distributed strategy which results in behaviors closed to what is obtained with a centralized approach. Recently, some decentralized receding horizon planners have been proposed. In (Dunbar and Murray, 2006), a distributed solution is provided for the rigid formation stabilization problem.

In (Kuwata et al., 2006), the navigation problem is solved through a coordination by leadership. Indeed, the robots update their trajectory sequentially. In (Defoort et al., 2007), a decentralized algorithm based on a coordination by adjustment is proposed to solve the navigation problem for MRS. However, the large amount of information exchanged between robots and the addition of several constraints make this strategy prohibitive when the number of vehicles increases.

One of other collision avoidance algorithm is potential field method, where an artificial potential function treats each robot as a charged particle that repels all the other entities (Latombe, 1991; De-Gennaro and Jadbabaie, 2006). However, most of them are only based on relative position information and do not consider coordination between cooperative robots.

In this paper, we proposed a practical decentralized scheme, based on a coordination by adjustment, for real time navigation of large-scale MRS. As illustrated in Fig. 1, the scheme consists of two parallel processes:

- a distributed receding horizon planner, in which each robot computes its own planned trajectory locally, for the coordination between cooperative robots,
- a reactive approach, which combines artificial potential fields and sliding mode control technique, for simultaneously tracking the planned trajectory while avoiding collision with unexpected entities (i.e. non cooperative entities).

The main advantages of the proposed strategy, especially for large-scale MRS, are the small amount of information exchanged between cooperative robots and the robustness.

The outline of this paper is as follows. In Section 2, the problem setup is described. In Section 3, the navigation algorithm is presented. Finally, numerical results illustrate the effectiveness of the strategy.

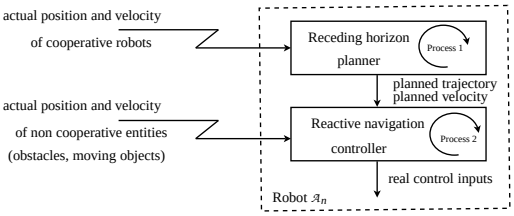


Figure 1: Proposed navigation algorithm.

2 PROBLEM STATEMENT

2.1 Dynamic Model of the Robots

Each robot $\mathcal{A}_n$ ($n \in (1,\dots,N)$ with $N \in \mathbb{N}$), shown in Fig. 2, is of unicycle-type. Its two fixed driving wheels of radius r_n, separated by $2\rho_n$, are independently controlled by two actuators (DC motors) and the passive wheel prevents the robot from tipping over as it moves on a plane. Its configuration is given by:

$$\boldsymbol{\eta}_n = [x_n, y_n, \theta_n]^T$$

where (x_n, y_n) is the position of its mass center C_n and θ_n is its orientation in the global frame.

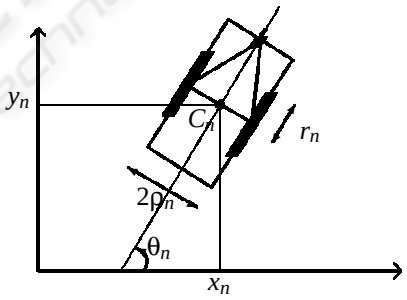


Figure 2: Unicycle-type robot.

The dynamic model of robot $\mathcal{A}_n$ is given as in (Do et al., 2004):

$$\dot{\boldsymbol{\eta}}_n = J(\boldsymbol{\eta}_n)\boldsymbol{z}_n \quad (1)$$

$$M_n\dot{\boldsymbol{z}}_n + D_n\boldsymbol{z}_n = \boldsymbol{\tau}_n \quad (2)$$

where

- M_n is a symmetric positive definite inertia matrix
- D_n is a symmetric damping matrix
- the transformation matrix $J(\boldsymbol{\eta}_n)$ is

$$J(\boldsymbol{\eta}_n) = \frac{r_n}{2}\begin{bmatrix} \cos\theta_n & \cos\theta_n \\ \sin\theta_n & \sin\theta_n \\ \rho_n^{-1} & -\rho_n^{-1} \end{bmatrix} \quad (3)$$

- $\boldsymbol{z}_n = [z_n^r, z_n^l]^T$ where z_n^r, z_n^l are the angular velocities of the right and left wheels. The relationship between $\boldsymbol{z}_n$ and the linear and angular velocities, denoted v_n, w_n, is

$$\begin{bmatrix} z_n^r \\ z_n^l \end{bmatrix} = B_n \begin{bmatrix} v_n \\ w_n \end{bmatrix} \text{ with } B_n = \frac{1}{r_n} \begin{bmatrix} 1 & \rho_n \\ 1 & -\rho_n \end{bmatrix} \quad (4)$$

- $\boldsymbol{\tau}_n = [\tau_n^r, \tau_n^l]^T$ where τ_n^r,τ_n^l are the control torques applied to the wheels of the robot

Remark 1. *System (1)-(2) is flat (see (Fliess et al., 1995) for details about flatness) since all system variables can be differentially parameterized by x_n, y_n as well as a finite number of their time derivatives. For instance, θ_n, v_n and w_n can be expressed as*

$$\theta_n = \arctan\frac{\dot{y}_n}{\dot{x}_n},\ v_n = \sqrt{\dot{x}_n^2 + \dot{y}_n^2},\ w_n = \frac{\ddot{y}_n\dot{x}_n - \ddot{x}_n\dot{y}_n}{\dot{x}_n^2 + \dot{y}_n^2}$$

Remark 2. *Speed $\boldsymbol{u}_n = [\dot{x}_n, \dot{y}_n]^T$ of $\mathcal{A}_n$ is restricted to lie in a closed interval $\mathcal{S}_n$*

$$\mathcal{S}_n = \left\{ \ \boldsymbol{u}_n \in \mathbb{R}^2 \quad | \quad \|\boldsymbol{u}_n\| \leq u_{n,max} \ \right\} \quad (5)$$

2.2 Assumptions and Control Objective

Assumption 1. *The following assumptions are made:*

- *$\mathcal{A}_n$ knows its position $\boldsymbol{p}_n = [x_n, y_n]^T$ and its velocity $\boldsymbol{u}_n = [\dot{x}_n, \dot{y}_n]^T$*
- *$\mathcal{A}_n$ has a physical safety area, which is centered at C_n with a radius R_n, and has a circular communication area which is also centered at C_n with a radius $\bar{R}_n$. Note that $\bar{R}_n$ is strictly larger than $R_n + R_j$, $j \in (1, \ldots, N), j \neq n$*
- *$\mathcal{A}_n$ broadcasts $(\boldsymbol{p}_n, \boldsymbol{u}_n)$ and receives $(\boldsymbol{p}_j, \boldsymbol{u}_j)$ broadcasted by other cooperative robots $\mathcal{A}_j$, in its communication area*
- *$\mathcal{A}_n$ can compute the relative position and velocity $(\boldsymbol{p}_{obs_i}, \boldsymbol{u}_{obs_i})$ of non cooperative entities within a given sensing range*
- *At the initial time $t_{ini} \geq 0$, each robot starts at a location outside of the safety areas of other entities*

The objective is to find the control inputs $\boldsymbol{\tau}_n$ for each robot $\mathcal{A}_n$ such that, under Assumption 1,

- $\mathcal{A}_n$ is stabilized toward its desired point $\boldsymbol{p}_{n,des}$, i.e.

$$\lim_{t\to\infty} \|\boldsymbol{p}_n(t) - \boldsymbol{p}_{n,des}\| = 0 \quad (6)$$

- collisions are avoided
- all computations are done on board in a decentralized cooperative way

Remark 3. *It should be noted that for collision avoidance, one can distinguish two kinds of entities, i.e.*

- *cooperative robots which are involved in a detected potential collision*
- *non cooperative entities which cannot cooperate in the collision avoidance process. They represent the moving objects and static obstacles.*

3 DISTRIBUTED ALGORITHM

In order to solve the multi-robots navigation problem, a decentralized algorithm combining two parallel processes is proposed. First, a receding horizon planner, in which each robot computes its own planned trajectory locally, achieves middle-term objectives, i.e. coordination between cooperative robots which are involved in a detected potential collision. Then, a reactive navigation controller is proposed to fulfill short-term objectives, i.e. trajectory tracking while taking into account non cooperative entities.

3.1 Conflicts and Collisions

Definition 1. *(conflict) A conflict occurs between two cooperative robots $\mathcal{A}_n$ and $\mathcal{A}_j$ at time $t_k \in \mathbb{R}^+$ if they are not in collision at t_k, but at some future time, a collision may occur.*

The following proposition, based on the well-known concept of velocity obstacle (Fiorini and Shiller, 1998), is useful to check the presence of conflicts.

Proposition 1. *Let us define for each pair $(\mathcal{A}_n, \mathcal{A}_j)$, the following variables depicted in Fig. 3:*

$$\begin{aligned} \beta_{nj}(t_k) &= \arg(\boldsymbol{u}_n(t_k) - \boldsymbol{u}_j(t_k)) - \arg(\boldsymbol{p}_j(t_k) - \boldsymbol{p}_n(t_k)) \\ \alpha_{nj}(t_k) &= \arcsin\left(\frac{R_n + R_j}{\|\boldsymbol{p}_j(t_k) - \boldsymbol{p}_n(t_k)\|}\right) \end{aligned} \quad (7)$$

A necessary and sufficient condition for no conflict between $\mathcal{A}_n$ and $\mathcal{A}_j$ at t_k is:

$$|\beta_{nj}(t_k)| \geq \alpha_{nj}(t_k) \quad (8)$$

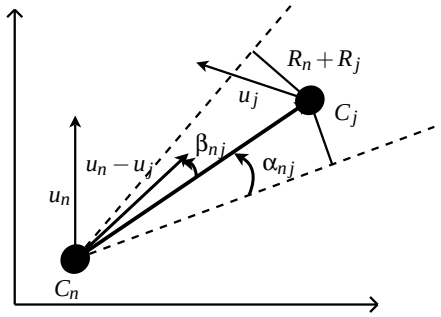


Figure 3: Velocity obstacle concept.

Definition 2. *(conflict subset) For each $\mathcal{A}_n$, the conflict subset $\mathcal{N}_n(t_k)$ at time $t_k \in \mathbb{R}^+$ is the set of all cooperative robots which are in the communication area of $\mathcal{A}_n$ and in conflict with $\mathcal{A}_n$.*

3.2 Receding Horizon Planner

The purpose of the distributed receding horizon planner is to decompose the overall problem into a family of simple receding horizon planning problems which are implemented in each robot $\mathcal{A}_n$.

In every problem, the same planning horizon $T_p \in \mathbb{R}^+$ and update period $T_c \in \mathbb{R}^+$ ($T_c < T_p$) are used. The receding horizon updates are

$$t_k = t_{ini} + (k-1)T_c, \;\; k \in \mathbb{N}^* \tag{9}$$

Remark 4. *During the initialization step, that is to say before robots move, we denote $t_0 = t_{ini}$.*

At each update t_k, robots in conflict exchange information about each others (position, velocity, ...). Then, in parallel, every robot $\mathcal{A}_n$ computes an anticipated trajectory, denoted $\widehat{\boldsymbol{p}}_j(t,t_k)$ and an anticipated velocity $\widehat{\boldsymbol{u}}_j(t,t_k)$, over the overall horizon, for all $\mathcal{A}_j$ belonging to $\mathcal{N}_n(t_k)$. These trajectories are obtained without taking the collision avoidance constraint into account. Therefore, by design, the anticipated trajectory is the same in every receding horizon planning problem in which it occurs. At last, in parallel, every robot $\mathcal{A}_n$ computes only its own planned trajectory $\boldsymbol{p}_n^*(t,t_k)$ and planned velocity $\boldsymbol{u}_n^*(t,t_k)$, over the planning horizon T_p, in order to integrate the collision avoidance between cooperative robots. From the planned trajectory and velocity associated to the planning horizon T_p, only the part which corresponds to the update horizon T_c is stored.

Remark 5. *Note that the first argument of $\boldsymbol{p}_n^*$, $\widehat{\boldsymbol{p}}_n$, $\boldsymbol{u}_n^*$ and $\widehat{\boldsymbol{u}}_n$ denotes time. The second argument is only added to distinguish at which receding horizon update the trajectory and velocity are computed.*

The collection of distributed receding horizon planning problems is formally defined as Problems 1-2 for each robot $\mathcal{A}_n$.

Problem 1. For each robot $\mathcal{A}_n$ and at any update t_k, $k \in \mathbb{N}$:
Given: the actual positions $\boldsymbol{p}_n(t_k)$, $\boldsymbol{p}_j(t_k)$ and the actual velocities $\boldsymbol{u}_n(t_k)$, $\boldsymbol{u}_j(t_k)$ of robot $\mathcal{A}_n$ and robots $\mathcal{A}_j$ belonging to $\mathcal{N}_n(t_k)$, respectively.
Find: the anticipated trajectory and velocity pairs $(\widehat{\boldsymbol{p}}_i(t,t_k), \widehat{\boldsymbol{u}}_i(t,t_k))$, $\forall i \in \{ i \in \mathbb{N} \mid \mathcal{A}_i \in \mathcal{N}_n(t_k) \cup \{\mathcal{A}_n\}\}$
subject to the following constraints:

$$\left\{ \begin{array}{lcl} \widehat{\boldsymbol{p}}_i(t_k,t_k) & = & \boldsymbol{p}_i(t_k) \\ \widehat{\boldsymbol{u}}_i(t_k,t_k) & = & \boldsymbol{u}_i(t_k) \\ \widehat{\boldsymbol{u}}_i(t,t_k) & \in & \mathcal{S}_i, \;\; \forall t \geq t_k \end{array} \right. \tag{10}$$

The anticipated trajectories are computed without taking the collision avoidance constraint into account. That is why, to integrate the path planning with local collision avoidance, the following problem is solved.

Problem 2. For each robot $\mathcal{A}_n$ and at any update t_k, $k \in \mathbb{N}$:
Given: the anticipated pairs $(\widehat{\boldsymbol{p}}_i(t,t_k), \widehat{\boldsymbol{u}}_i(t,t_k))$, $\forall i \in \{ i \in \mathbb{N} \mid \mathcal{A}_i \in \mathcal{N}_n(t_k) \cup \{\mathcal{A}_n\}\}$.
Find: the planned trajectory and velocity pairs $(\boldsymbol{p}_n^*(t,t_k), \boldsymbol{u}_n^*(t,t_k))$ that minimizes

$$\int_{t_{k+1}}^{t_{k+1}+T_p} \left(a_n \sum_j \widehat{U}_{nj,rep}(t) + \| \boldsymbol{p}_n^*(t,t_k) - \widehat{\boldsymbol{p}}_n(t,t_k) \| \right) dt \tag{11}$$

subject to the following constraints:

$$\left\{ \begin{array}{lcl} \boldsymbol{p}_n^*(t_{k+1},t_k) & = & \boldsymbol{p}_n^*(t_{k+1},t_{k-1}) \\ \boldsymbol{u}_n^*(t_{k+1},t_k) & = & \boldsymbol{u}_n^*(t_{k+1},t_{k-1}) \\ \boldsymbol{u}_n^*(t,t_k) & \in & \mathcal{S}_n, \; \forall t \in \left[t_{k+1}, t_{k+1}+T_p\right] \end{array} \right. \tag{12}$$

where

$$\begin{array}{rcl} \widehat{U}_{nj,rep}(t) & = & \left\{ \begin{array}{ll} 0 & \text{if } \widehat{\rho}_{nj}(t) \geq b_n \\ \frac{1}{2}\left(\frac{1}{\widehat{\rho}_{nj}(t)} - \frac{1}{b_n}\right)^2 & \text{else} \end{array} \right. \\ \widehat{\rho}_{nj}(t) & = & \| \boldsymbol{p}_n^*(t,t_k) - \widehat{\boldsymbol{p}}_j(t,t_k) \| - (R_n + R_j) \end{array} \tag{13}$$

a_n and b_n are strictly positive factors which can vary among robots to reflect differences in aggressiveness ($a_n < 1$, $b_n \ll 1$) and shyness ($a_n > 1$, $b_n \gg 1$).

One can note that the first part of cost (11) is designed to enforce the collision avoidance between cooperative robots. The cost term $\| \boldsymbol{p}_n^*(t,t_k) - \widehat{\boldsymbol{p}}_n(t,t_k) \|$ in (11) is a way of penalizing the deviation of the planned trajectory $\boldsymbol{p}_n^*(t,t_k)$ from the anticipated trajectory $\widehat{\boldsymbol{p}}_n(t,t_k)$, which is the trajectory that other robots rely on. In previous work, this term was incorporated into the decentralized receding horizon planner as a constraint (Defoort et al., 2007). The formulation presented here is an improvement over this past formulation, since the penalty yields an optimization problem that is much easier to solve.

Remark 6. *One can note that constraints (12) which guarantee the continuity of the planned trajectory and velocity need $\boldsymbol{p}_n^*(t_{k+1},t_{k-1})$ and $\boldsymbol{u}_n^*(t_{k+1},t_{k-1})$ computed in the previous step. Therefore, the proposed planner is not able to reject external disturbances or inherent discrepancies between the model and the real process. However, it takes the real time constraint into account. Indeed, each robot has a limited time to plan its trajectory. The time allocated to make its decision depends on its perception sensors, its computation delays and is less than the update period T_c (see Fig. 4).*
The discussed claim for robustness in trajectory tracking will be achieved hereafter.

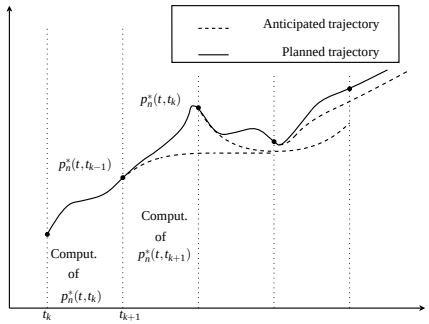


Figure 4: Implementation of the receding horizon planner.

Remark 7. *A compromise must be done between reactivity and computation time. Indeed, the planning horizon must be sufficiently small in order to have good enough results in terms of computation time. However, it must be higher than the update period to guarantee enough reactivity.*

Remark 8. *To numerically solve Problems 1-2, a nonlinear trajectory generation algorithm (Defoort et al., 2009) is applied. It is based on finding trajectory curves in a lower dimensional space and parameterizing these curves by B-splines. A constrained feasible sequential quadratic optimization algorithm is used to find the B-splines coefficients that optimize the performance objective while respecting the constraints.*

3.3 Reactive Navigation Controller

Hereafter, a reactive approach, which combines artificial potential fields and sliding mode control technique, for simultaneously tracking the planned trajectory while avoiding collision with unexpected entities (i.e. non cooperative entities), is proposed.

Since the robot dynamics (1)-(2) is of strict feedback systems (see (Krstic et al., 1995) for details about strict feedback systems) with respect to the robot linear and angular velocities (i.e. v_n and w_n), a backstepping procedure is used to design the control input $\boldsymbol{\tau}_n$. That is why the control design is divided into two main steps.

3.3.1 Step 1 based on Artificial Potential Fields

Let us introduce the following notations:

$$\begin{cases} \theta_{ne} &= \theta_n - \gamma_{\theta_n} \\ v_{ne} &= v_n - \gamma_{v_n} \end{cases} \tag{14}$$

where γ_{θ_n} and γ_{v_n} are auxiliary variables used to avoid collisions. Replacing expressions (14) into the first two equations of (1) and using (4) yield:

$$\dot{\boldsymbol{p}}_n = \begin{bmatrix} \cos\gamma_{\theta_n} \\ \sin\gamma_{\theta_n} \end{bmatrix} \gamma_{v_n} + \Delta_{1n} + \Delta_{2n} \tag{15}$$

with $\Delta_{1n} = \gamma_{v_n} \begin{bmatrix} (\cos\theta_{ne}-1)\cos\gamma_{\theta_n} - \sin\theta_{ne}\sin\gamma_{\theta_n} \\ \sin\theta_{ne}\cos\gamma_{\theta_n} + (\cos\theta_{ne}-1)\sin\gamma_{\theta_n} \end{bmatrix}$ and $\Delta_{2n} = v_{ne} \begin{bmatrix} \cos\theta_n \\ \sin\theta_n \end{bmatrix}$.

The objective is to design the auxiliary variables γ_{v_n} and γ_{θ_n} such that robot $\mathcal{A}_n$ robustly tracks its planned trajectory $\boldsymbol{p}_n^*$ while avoiding unexpected collisions. Here, artificial potential functions are used in order to design an attractive force between the robot and its planned trajectory and a repulsive force to avoid collisions.

In conventional potential field method (Latombe, 1991), the planned robot velocity $\boldsymbol{u}_n^*$ is assumed to be zero and the obstacle velocity $\boldsymbol{u}_{obs_i}$ is not considered. However, to make robot $\mathcal{A}_n$ track the planned trajectory among moving obstacles, velocities $\boldsymbol{u}_n^*$ and $\boldsymbol{u}_{obs_i}$ play key roles. This issue will be addressed by extending the results given in (Huang, 2009). Let us consider the conventional potential function (Latombe, 1991):

$$U_n = U_{n,att} + U_{n,rep} \tag{16}$$

where $U_{n,att}$ and $U_{n,rep}$ are, respectively, the attractive potential defined to track the planned trajectory $\boldsymbol{p}_n^*$ and the repulsive potential related to collision avoidance, specified as follows:

- The attractive potential is designed such that it puts penalty on the tracking error and is equal to zero when the robot is at its desired position, i.e.

$$U_{n,att} = \frac{1}{2}\left\|\boldsymbol{p}_n - \boldsymbol{p}_n^*\right\|^2 \tag{17}$$

- The repulsive potential is designed such that it equals to infinity when a collision occurs with $\mathcal{A}_n$ and decreases according to the relative distance between $\mathcal{A}_n$ and an obstacle, i.e.

$$U_{n,rep} = c_n \sum_i U_{ni,rep} \tag{18}$$

with

$$U_{ni,rep} = \begin{cases} 0 & \text{if } \rho_{ni} \geq d_n \\ \frac{1}{2}\left(\frac{1}{\rho_{ni}} - \frac{1}{d_n}\right)^2 & \text{else} \end{cases} \tag{19}$$

where

ρ_{ni} is the minimum distance between robot $\mathcal{A}_n$ and the obstacle i. c_n and d_n are strictly positive factors which have similar properties as a_n and b_n.

Proposition 2. *If the errors θ_{ne} and v_{ne} are asymptotically stable, $\mathcal{A}_n$ robustly tracks its planned trajectory $\boldsymbol{p}_n^*$ while avoiding collisions using the auxiliary variables:*

$$\begin{aligned} \gamma_{v_n} = {} & [(\|\boldsymbol{u}_n^*\|\cos(\theta_n^* - \psi_n) - c_n \textstyle\sum_i \xi_{ni}\|\boldsymbol{u}_{obs_i}\|\cos(\theta_{obs_i} - \psi_{ni}) \\ & + \|\boldsymbol{p}_n - \boldsymbol{p}_n^*\|)^2 + \|\boldsymbol{u}_n^*\|^2 \sin^2(\theta_n^* - \bar{\psi}_n)]^{0.5} \\ \gamma_{\theta_n} = {} & \bar{\psi}_n + \arcsin\left(\frac{\|\boldsymbol{u}_n^*\|\sin(\theta_n^* - \bar{\psi}_n)}{\gamma_{v_n}}\right) \end{aligned} \tag{20}$$

with

$$\begin{array}{lcl}
\theta_n^* & = & \arg(\boldsymbol{u}_n^*) \\
\theta_{obs_i} & = & \arg(\boldsymbol{u}_{obs_i}) \\
\psi_n & = & \arg(\boldsymbol{p}_n^* - \boldsymbol{p}_n) \\
\psi_{ni} & = & \arg(\boldsymbol{p}_{obs_i} - \boldsymbol{p}_n) \\
\bar{\psi}_n & = & \arctan\left(\frac{\sin\psi_n - c_n\sum_i \xi_{ni}\sin\psi_{ni}}{\cos\psi_n - c_n\sum_i \xi_{ni}\cos\psi_{ni}}\right) \\
\xi_{ni} & = & \begin{cases} 0 & \text{if } \rho_{ni} \geq d_n \\ \left(\frac{1}{\rho_{ni}} - \frac{1}{d_n}\right)\frac{1}{(\rho_{ni})^2}\frac{1}{\|\boldsymbol{p}_n - \boldsymbol{p}_n^*\|} & \text{else} \end{cases}
\end{array}$$

Proof. Let us differentiate U_n with respect to time in equation (16), i.e.:

$$\dot{U}_n = \dot{U}_{n,att} + \dot{U}_{n,rep} \tag{21}$$

Substituting (20) into (21) yields after some geometric manipulations:

$$\begin{array}{rcl}
\dot{U}_n & = & \|\boldsymbol{p}_n^* - \boldsymbol{p}_n\|(\|\boldsymbol{u}_n^*\|\cos(\theta_n^* - \psi_n) - \gamma_{v_n}\cos(\gamma_{\theta_n} - \psi_n) \\
 & & -c_n\sum_i \xi_{ni}\left(\|\boldsymbol{u}_{obs_i}\|\cos(\theta_{obs_i} - \psi_{ni}) + \gamma_{v_n}\cos(\gamma_{\theta_n} - \psi_{ni})\right)) \\
 & & + \left((\boldsymbol{p}_n - \boldsymbol{p}_n^*)^T - c_n\sum_i \xi_{ni}\frac{\|\boldsymbol{p}_n - \boldsymbol{p}_n^*\|}{\|\boldsymbol{p}_n - \boldsymbol{p}_{obs_i}\|}(\boldsymbol{p}_n - \boldsymbol{p}_{obs_i})^T\right)(\Delta_{1n} + \Delta_{2n}) \\
 & = & \|\boldsymbol{p}_n^* - \boldsymbol{p}_n\|(\|\boldsymbol{u}_n^*\|\cos(\theta_n^* - \psi_n) - \gamma_{v_n}\cos(\gamma_{\theta_n} - \bar{\psi}_n) \\
 & & -c_n\sum_i \xi_{ni}\left(\|\boldsymbol{u}_{obs_i}\|\cos(\theta_{obs_i} - \psi_{ni})\right)) \\
 & & + \left((\boldsymbol{p}_n - \boldsymbol{p}_n^*)^T - c_n\sum_i \xi_{ni}\frac{\|\boldsymbol{p}_n - \boldsymbol{p}_n^*\|}{\|\boldsymbol{p}_n - \boldsymbol{p}_{obs_i}\|}(\boldsymbol{p}_n - \boldsymbol{p}_{obs_i})^T\right)(\Delta_{1n} + \Delta_{2n}) \\
 & = & \|\boldsymbol{p}_n^* - \boldsymbol{p}_n\|(\|\boldsymbol{u}_n^*\|\cos(\theta_n^* - \psi_n) - \sqrt{\gamma_{v_n}^2 - \|\boldsymbol{u}_n^*\|^2\sin^2(\theta_n^* - \bar{\psi}_n)} \\
 & & -c_n\sum_i \xi_{ni}\left(\|\boldsymbol{u}_{obs_i}\|\cos(\theta_{obs_i} - \psi_{ni})\right)) \\
 & & + \left((\boldsymbol{p}_n - \boldsymbol{p}_n^*)^T - c_n\sum_i \xi_{ni}\frac{\|\boldsymbol{p}_n - \boldsymbol{p}_n^*\|}{\|\boldsymbol{p}_n - \boldsymbol{p}_{obs_i}\|}(\boldsymbol{p}_n - \boldsymbol{p}_{obs_i})^T\right)(\Delta_{1n} + \Delta_{2n})
\end{array}$$

Assuming that the errors θ_{ne} and v_{ne} are asymptotically stable (i.e. $\Delta_{1n} = \Delta_{2n} = 0$), one can get from (20):

$$\dot{U}_n \leq -\|\boldsymbol{p}_n^* - \boldsymbol{p}_n\|^2 \tag{22}$$

Since $U_n \geq 0$ and $\dot{U}_n \leq 0$, U_n is bounded. That is why $\mathcal{A}_n$ robustly tracks its planned trajectory $\boldsymbol{p}_n^*$ while avoiding collisions. □

3.3.2 Step 2 based on Sliding Mode Technique

Now, the objective is to force the motion of robot $\mathcal{A}_n$ such that the errors θ_{ne} and v_{ne} are asymptotically stable. The proposed strategy is based on the so-called second order sliding mode control (SMC) approach. The SMC methodology (Utkin et al., 1999) is chosen because it is a robust technique to control nonlinear systems operating under uncertainty conditions (Fridman and Levant, 2002). Furthermore, second order SMC can reduce the chattering phenomenon (high frequency vibrations of the controlled system which degrade the performances). Indeed, instead of influencing the first sliding variable time derivative, the signum function acts on its second time derivative. This method can also achieve a better convergence accuracy with respect to discrete sampling time than conventional SMC (see (Fridman and Levant, 2002) for a survey).

Let us apply to system (1)-(2) the following preliminary feedback:

$$\bar{\boldsymbol{\tau}}_n = (M_n B_n)^{-1}(\boldsymbol{\tau}_n - D_n \boldsymbol{z}_n) \tag{23}$$

where $\bar{\boldsymbol{\tau}}_n = [\bar{\tau}_{1n}, \bar{\tau}_{2n}]^T$ is the auxiliary control input. Thus, system (1)-(2) can be expressed as follows:

$$\dot{\boldsymbol{\eta}}_n = J(\boldsymbol{\eta}_n)B_n\begin{bmatrix} v_n \\ w_n \end{bmatrix} \tag{24}$$

$$\begin{bmatrix} \dot{v}_n \\ \dot{w}_n \end{bmatrix} = \bar{\boldsymbol{\tau}}_n \tag{25}$$

Since the relative degree of system (24)-(25) with respect to the sliding variable v_{ne} is only one, a dynamic extension is done before designing the control (see (Isidori, 1989) for further details). Thus, an integrator chain is added on the input variable $\bar{\tau}_{1n}$.

There are several algorithms to ensure the finite time stabilization of the sliding variables θ_{ne} and v_{ne} towards the origin. Among them, the sampled twisting algorithm (Fridman and Levant, 2002) has been developed for systems with relative degree two. This algorithm provides good convergence accuracy and robustness properties. It does not require the knowledge of the time derivative of the sliding variables and takes into account some practical constraints such as the sampling of the measurement and the control.

Proposition 3. *Consider system (1)-(2). The errors θ_{ne} and v_{ne} are stable in finite time under the nonlinear controller defined in (23) where*

$$\begin{array}{l}
\bar{\tau}_{1n} = \begin{cases} -\lambda_{1,M}\,\mathrm{sign}(\theta_{ne}) & \text{if } \theta_{ne}\Delta_{\theta_{ne}} > 0 \\ -\lambda_{1,m}\,\mathrm{sign}(\theta_{ne}) & \text{if } \theta_{ne}\Delta_{\theta_{ne}} \leq 0 \end{cases} \\
\bar{\tau}_{2n} = \begin{cases} -\lambda_{2,M}\,\mathrm{sign}(v_{ne}) & \text{if } v_{ne}\Delta_{v_{ne}} > 0 \\ -\lambda_{2,m}\,\mathrm{sign}(v_{ne}) & \text{if } v_{ne}\Delta_{v_{ne}} \leq 0 \end{cases}
\end{array} \tag{26}$$

with

$$\begin{array}{l}
\Delta_{\theta_{ne}} = \begin{cases} 0 & \text{if } k = 0 \\ \theta_{ne}(kT_s) - \theta_{ne}((k-1)T_s) & \text{else} \end{cases} \\
\Delta_{v_{ne}} = \begin{cases} 0 & \text{if } k = 0 \\ v_{ne}(kT_s) - v_{ne}((k-1)T_s) & \text{else} \end{cases}
\end{array} \tag{27}$$

T_s is the sampling period, $k \in \mathbb{N}$ is related to the time of the process and $\lambda_{i,m}$, $\lambda_{i,M}$, $i = 1,2$ are positive constants high enough to enforce the sliding motion.

Proof. It can be shown that this controller ensures a finite time convergence of the trajectories onto the manifold $\{v_{ne} = \dot{v}_{ne} = 0\}$ and $\{\theta_{ne} = \dot{\theta}_{ne} = 0\}$ (see (Fridman and Levant, 2002) for further details). Hence, the application of the control input (26) results in the robust finite time stabilization of θ_{ne} and v_{ne}. □

Remark 9. *We would like to emphasize that although not explicitly considered here the procedure based on sliding mode control guarantees proper behavior even in the presence of uncertainties in the mass and inertia of the robots and additive disturbances to the linear and angular velocities which constitute very realistic assumptions.*

Once the sliding mode occurs on all the surfaces (which happens in finite time), based on Proposition 2, the global control objectives, defined in Section 2.2, are fulfilled.

Some specific advantages of the proposed decentralized algorithm are enumerated below:

- robustness with respect to uncertainties and disturbances (sliding mode controller),
- reactivity (potential field functions),
- low communication bandwidth, i.e. small amount of information is locally exchanged,
- reduction of deadlocks due to local minima in potential field (anticipation and coordination mechanism through the receding horizon planner).

4 SIMULATION RESULTS

This section demonstrates the performance of the proposed decentralized algorithm. The following simulations showcase two different scenarios for which the environment is partially known (i.e. the range of sensors of each robot is of radius $1.5m$).

The main parameters of the robots are: $\forall n$, $R_n = 0.25m$, $\bar{R}_n = 4m$ and $u_{n,max} = 1m/s$. For the decentralized algorithm, the following parameters are used: $T_p = 3s$, $T_c = 0.5s$, $a_n = c_n = 1$, $b_n = 2$, $d_n = 0.5$, $\lambda_{1,M} = \lambda_{2,M} = 10$, $\lambda_{1,m} = \lambda_{2,m} = 1$ and $T_s = 0.01s$.

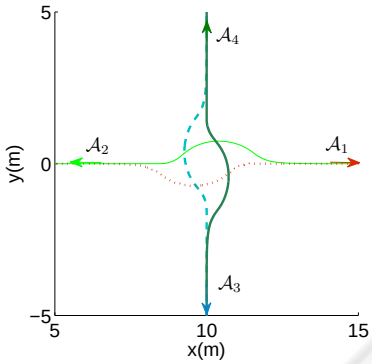
4.1 Scenario 1: Crossing

In this scenario, there are four robots ($N = 4$) starting at $\boldsymbol{p}_1(t_{ini}) = [5,0]^T$, $\boldsymbol{p}_2(t_{ini}) = [15,0]^T$, $\boldsymbol{p}_3(t_{ini}) = [10,5]^T$ and $\boldsymbol{p}_4(t_{ini}) = [10,-5]^T$ respectively, with velocities equal to zero. These robots must cross each other in order to reach their desired configuration $\boldsymbol{p}_{1,des} = [15,0]^T$, $\boldsymbol{p}_{2,des} = [5,0]^T$, $\boldsymbol{p}_{3,des} = [10,-5]^T$ and $\boldsymbol{p}_{4,des} = [10,5]^T$. One can note that this problem is not trivial due to its symmetry properties.

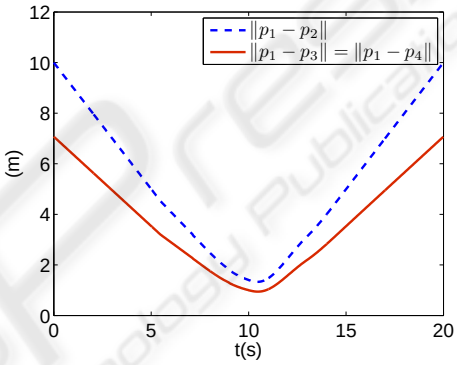
The simulation results are given in Fig. 5. One can see that each robot modifies its trajectory in order to avoid collision. Figure 5(b) depicts the evolution of the distance between robots. Since it is higher than $0.5m$, the collision avoidance is guaranteed.

4.2 Scenario 2: Reconfiguration with Collision Avoidance

In this scenario, a swarm of five robots ($N = 5$) reconfigures its geometric shape (from "linear" to "triangular") while avoiding collisions with obstacles. The



(a)



(b)

Figure 5: Four vehicles simulation: (a) Robot trajectories. (b) Relative distances between $\mathcal{A}_1$ and other robots.

proposed decentralized controller has only a limited knowledge of the obstacles (initially unknown). It simply keeps the robots spaced out using the proposed potential field technique. The five robots make decisions in order to avoid collisions. One can note that the number of potential conflicts is high.

One can see in Fig. 6 that under the proposed decentralized algorithm, the robots meet the objective defined in Section 2.2. Note that the radius of obstacles is increased by 0.25m (dotted lines around obstacles) to take the size of robots into account.

5 CONCLUSIONS

A new distributed strategy for the navigation of multiple autonomous robots is presented. The proposed scheme combines a decentralized receding horizon motion planner to satisfy middle-term objectives (coordination between cooperative robots) with a fast navigation controller based on artificial potential fields and sliding mode control technique to satisfy short-term objectives (collision avoidance and

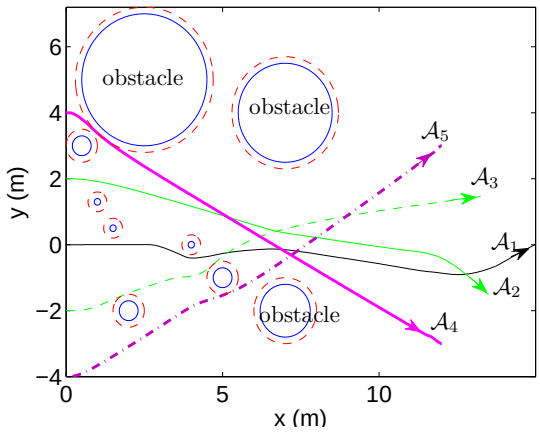


Figure 6: Collision avoidance of five robots.

trajectory tracking). The fact that there is no leader increases the security and the robustness of the missions. Simulation studies are provided in order to show the effectiveness of the proposed approach.

Experimental testing on WifiBot is under way. In the future, it is planned to design real time observers to estimate the relative velocities between robots.

REFERENCES

Das, A., Fierro, R., Kumar, V., Ostrowski, J., Spetzer, J., and Taylor, C. (2002). A vision-based formation control framework. *IEEE T. Robotic Autom.*, 18(5):pp. 813–825.

De-Gennaro, M. and Jadbabaie, A. (2006). Formation control for a cooperative multi-agent system using decentralized navigation functions. In *American Control Conf.*

Defoort, M., Floquet, T., Kokosy, A., and Perruquetti, W. (2007). Decentralized robust control for multi-vehicle navigation. In *European Control Conf.*

Defoort, M., Palos, J., Kokosy, A., Floquet, T., and Perruquetti, W. (2009). Performance based reactive navigation for nonholonomic mobile robots. *Robotica*, 27(2):pp. 281–290.

Do, K., Jiang, Z., and Pan, J. (2004). A global output-feedback controller for simultaneous tracking and stabilization of unicycle-type mobile robots. *IEEE T. Automat. Contr.*, 20(3):pp. 589–594.

Dunbar, W. and Murray, R. (2002). Model predictive control of coordinated multi-vehicle formation. In *IEEE Conf. on Decision and Control*.

Dunbar, W. and Murray, R. M. (2006). Distributed receding horizon control for multi-vehicle formation stabilization. *Automatica*, 42(4):pp. 549–558.

Fiorini, P. and Shiller, Z. (1998). Motion planning in dynamic environments using velocity obstacles. *Int. J. Robot. Res.*, 17(7):pp. 760–772.

Fliess, M., Levine, J., Martin, P., and Rouchon, P. (1995). Flatness and defect of nonlinear systems: introductory theory and examples. *Int. J. Control*, 61(6):pp. 1327–1361.

Fridman, L. and Levant, A. (2002). Higher order sliding mode modes. *Sliding mode control in Engineering, Ed W. Perruquetti, J. P. Barbot*, pages 53–101.

Huang, L. (2009). Velocity planning for a mobile robot to track a moving target - a potential field approach. *Robot. Auton. Syst.*, 57(1):pp. 55–63.

Isidori, A. (1989). *Nonlinear control systems*. Springer, New York, 2nd edition.

Krstic, M., Kanellakopoulos, I., and Kokotovic, P. (1995). *Nonlinear and adaptative control design*. Wiley, N. Y.

Kuchar, J. and Yang, L. (2000). A review of conflict detection and resolution modeling methods. *IEEE T. Intell. Transp.*, 1(4):pp. 179–189.

Kuwata, Y., Richards, A., Schouwenaraars, T., and How, J. (2006). Decentralized robust receding horizon control. In *American Control Conf.*

Latombe, J. (1991). *Robot Motion Planning*. Kluwer Academic Publishers, Norwell, MA.

Mayne, D., Rawlings, J., Rao, C., and Scokaert, P. (2000). Constrained model predicitive control: Stability and optimality. *Automatica*, 36(6):pp. 789–814.

Pallatino, L., Scordio, V., Bicchi, A., and Frazoli, E. (2007). Decentralized cooperative policy for conflict resolution in multivehicle systems. *IEEE T. Robot.*, 23(6):pp. 1170–1183.

Tomlin, C., Pappas, G., and Sastry, S. (1998). Conflict resolution for air traffic management: a study in multiagent hybrid systems. *IEEE T. Automat. Contr.*, 43(4):pp. 509–521.

Utkin, V., Guldner, J., and Shi, J. (1999). *Sliding Modes Control in Electromechanical Systems*. Taylor and Francis, New York.