Real-time Optimization of Trajectories that Guarantee the Rendezvous of Mobile Robots

Sven Gowal and Alcherio Martinoli

***Abstract*— Since the 1960s, *consensus* problems have puzzled the minds of many researchers in fields ranging from computer science to information aggregation. In this work, although specifically addressing the *rendezvous* problem for a team of mobile robots, we develop a methodology that can also be applied to other consensus problems, where optimality is important and where non-holonomicity characterizes the system at hand. In particular, we consider a group of differential-wheeled robots endowed with noisy relative positioning capabilities. We develop a distributed, real-time optimization method based on a receding horizon controller that minimizes a user-defined cost whilst guaranteeing the rendezvous. Finally, we perform experiments on real robots to confirm the validity of our approach.**

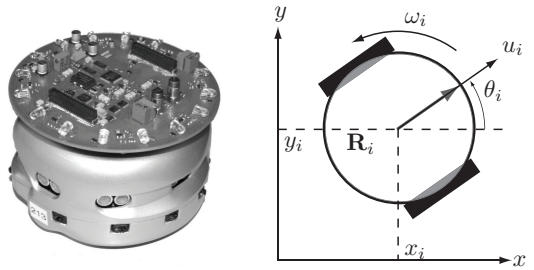
I. Introduction

The distributed convergence of mobile robots to a common location in space is a line of research that started with the *flock centering* rule of Reynolds [35] in 1987. Indeed, this ability to meet or to rendezvous has many practical applications such as the docking of spacecrafts [15] or cooperative aerial surveillance [2]. It is worth noting that the rendezvous problem itself is part of a wider range of problems better known as consensus problems. Although this paper specifically treats the rendezvous of differential-wheeled robots, its general concept may be applied to other fields including, but not restricted to, formation control [14], flocking [10] or attitude alignment [33].

To the best of our knowledge, it is only in 1999 that the convergence of mobile robots to a common location in space was studied [1]. It was later extended for both synchronous and asynchronous cases by Lin et al. [22, 23]. Their robots, however, were holonomic and, as such, yielded simpler control laws and tractable convergence properties. Solving the rendezvous with nonholonomic agents is more complex, and proving the convergence property can be difficult. Many contributions employ *feedback linearization* to design relaxed control laws that recreate the holonomic properties [21, 34]; others create algorithms that are very specific to their application needs [8, 9]; but all of them rely on deterministic assumptions both in terms of control and input.

Recently, our previous work [18] incorporated insights gathered by research on the *probabilistic consensus* problem [3, 4, 12] to extract rules guaranteeing that differential-wheeled robots achieve the rendezvous under noisy measurements. This approach and all previous approaches to solve the rendezvous problem on nonholonomic mobile robots rely heavily on strict time-invariant controllers that yield potentially poor trajectories without consideration of neither actuation constraints nor the energy spent. In this work, we address this issue by allowing a user to specify a local cost function which has to be minimized whilst additional constraints guarantee that the rendezvous happens even under noisy perception.

Sven Gowal and Alcherio Martinoli are with the Distributed Intelligent Systems and Algorithms Laboratory, School of Architecture, Civil and Environmental Engineering, École Polytechnique Fédérale de Lausanne. svenadrian.gowal@epfl.ch, alcherio.martinoli@epfl.ch



(a) (b)

Fig. 1. (a) A Khepera III robot with a range and bearing module attached. This range and bearing platform features sixteen infrared light emitting diodes and eight infrared light sensors. (b) The kinematic model of a differential-wheeled robot $\mathbf{R}_i$.

On another front, a great body of literature starting with Meschler [27] in 1963 focuses on the optimization of the rendezvous maneuver, and although efforts to decentralize the optimization approach using communication between agents have been made [26], many works remain centralized [6, 28] and thus need global knowledge of the system. We can also observe that most work, including [26], use a predefined cost function (e.g., time optimality [27] or fuel optimality [5]), and leave no design choices to the user. Finally, we note that, to date, no contribution has addressed the generation of real-time, optimal (or even sub-optimal) rendezvous maneuvers on real mobile robots performing noisy positioning observations — neither from a theoretical nor from an experimental point of view. In this work, we address this gap.

A. Problem Formulation

We have a team of N point-sized, differential-wheeled robots $\mathbf{R}_1, \ldots, \mathbf{R}_N$ driven by the kinematic equations:

$$\begin{cases} \dot{x}_i &= u_i \cos\theta_i \\ \dot{y}_i &= u_i \sin\theta_i \\ \dot{\theta}_i &= \omega_i \end{cases} \tag{1}$$

where $\mathbf{u}_i = [u_i, \omega_i]^\mathsf{T}$ is the vector of control inputs, with u_i the linear translational speed and ω_i the ro-

tational speed, and the vector $\mathbf{x}_i = [x_i, y_i, \theta_i]^\mathsf{T}$ defines the absolute pose or state of the robot $\mathbf{R}_i$, as shown in Figure 1(b). The state of the all robots is stored in the vector $\mathbf{x} = [x_1, y_1, \theta_1, \ldots, x_N, y_N, \theta_N]^\mathsf{T}$ and the control inputs applied to the system are denoted by $\mathbf{u} = [u_1, \omega_1, \ldots, u_N, \omega_N]^\mathsf{T}$.

Each robot $\mathbf{R}_i$ has a set of neighbors $\mathcal{N}_i$ containing all robots $\mathbf{R}_j$ such that it can measure the relative range e_{ij} and bearing α_{ij}. Note that unless stated otherwise, u_i means $u_i(t)$, ω_i means $\omega_i(t)$, e_{ij} means $e_{ij}(t)$ and α_{ij} means $\alpha_{ij}(t)$. Range and bearing measurements may be affected by noise such that each observation $z_{ij}(t)$ of $\mathbf{R}_j$ at time t is defined by the vector

$$z_{ij}(t) = \begin{bmatrix} \tilde{e}_{ij}(t) \\ \tilde{\alpha}_{ij}(t) \end{bmatrix} = \begin{bmatrix} e_{ij}(t) \\ \alpha_{ij}(t) \end{bmatrix} + \epsilon_z \tag{2}$$

where ϵ_z is a random noise vector. Hence at time t, a robot $\mathbf{R}_i$ gathers an observation list $Z_i(t) = \{z_{ij}(t) | \mathbf{R}_j \in \mathcal{N}_i\}$.

Our goal is to drive all robots to the same meeting point. This rendezvous maneuver should be performed in real-time, and optimally given a user-defined cost $\mathcal{J}(\mathbf{u})$. A user may design the functional $\mathcal{J}$ as he/she wishes as long as it can be written as a sum $\sum_i^N \mathcal{J}_i(\cdot)$ where $\mathcal{J}_i(\cdot)$ should only depend on values directly measurable (either through sensors or communication) or calculable by each individual robot $\mathbf{R}_i$. Without loss of generality, throughout this paper, we will use the Bolza forms $\mathcal{J}(\cdot) = \int L(\cdot)\mathrm{d}t + V(\cdot)$ and $\mathcal{J}_i(\cdot) = \int L_i(\cdot)\mathrm{d}t + V_i(\cdot)$ where $L(\cdot)$, $L_i(\cdot)$ are immediate cost functions and $V(\cdot)$, $V_i(\cdot)$ are terminal cost functions (also called *salvage terms*).

B. Graph Theory

Our network of robots can be seen as a graph containing N elements and can be described by the tuple $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where

- $\mathcal{V} = \{\mathbf{R}_1, \ldots, \mathbf{R}_N\}$ is the vertex set and
- $\mathcal{E} = \{\mathsf{e}_k | \mathsf{e}_k = (\mathbf{R}_i, \mathbf{R}_j) \implies \mathbf{R}_j \in \mathcal{N}_i\}$ is the edge set.

A graph is said to be *connected* if for every vertex pair there exists a path from one vertex to the other. On a graph $\mathcal{G}$, one can define an adjacency matrix $\mathbf{A}$ such that each of its elements a_{ij} is defined as

$$a_{ij} = \begin{cases} 1 & \text{if } \mathsf{e}_k = (\mathbf{R}_i, \mathbf{R}_j) \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases}.$$

A graph is *symmetric* if $a_{ij} = a_{ji}$.

In this work, we will consider a symmetric and connected graph defined by $\mathbf{A}$. First, the notion of symmetric connection in a group of homogeneous robots does make sense in reality where, often, if a robot $\mathbf{R}_i$ can observe another robot $\mathbf{R}_j$, then $\mathbf{R}_j$ can observe $\mathbf{R}_i$. Second, there must exist a path from each robot to any other robot for the rendezvous to happen globally, hence the graph must be connected. If the graph is composed of several independent units such that it is not connected, then each unit (or subgraph) can only achieve the rendezvous locally.

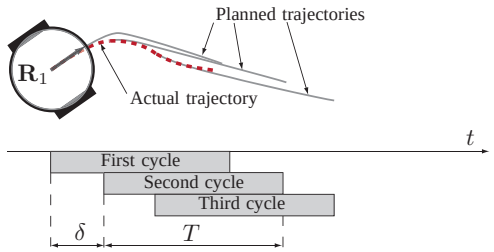


Fig. 2. Receding horizon trajectories: $\mathbf{R}_1$ plans an initial trajectory for the next T seconds and executes that trajectory *blindly* for the first δ seconds; at that point, $\mathbf{R}_1$ plans a new trajectory (and so on).

II. Centralized Receding Horizon Control

To solve our optimization problem (i.e., minimizing $\mathcal{J}(\cdot)$ whilst guaranteeing the rendezvous, as seen in Section I-A), we will rely on what is known as receding horizon control (RHC) [16]. RHC carries many names such as model predictive control (MPC) or real-time optimization (RTO). It is an advanced method, widely used in industry, that has the ability to use the available information on the system at hand to control it sub-optimally under a user-defined cost. Although its requirements in terms of computing power are high [30], it has found many successful applications, in particular when the underlying system to control has slow dynamics (i.e., in the order of minutes or seconds). The recent advances in computing power have in part alleviated this issue, but RHC reaches its limits when the underlying system is nonlinear and fast-changing.

To ease our discussion, we first introduce a centralized RHC policy that solves our problem (i.e., assuming global knowledge) without noisy measurements (i.e., we have access to all measurable ranges e_{ij} and bearings α_{ij}). Subsequently, Section III will introduce an equivalent decentralized control that reduces the computing requirements linked to RHC and generates real-time maneuvers that guarantee the rendezvous even with noisy measurements.

A. Standard Policy

RHC is an optimization-based control that uses online, optimal trajectory generation. The general idea is to plan a feasible and sub-optimal trajectory over a finite time horizon T and to control the system (i.e., the robots) to follow this trajectory over a period δ ($0 < \delta \leq T$). After δ seconds, a new trajectory is recomputed from the current position until time $\delta + T$ and this trajectory is followed until time 2δ. This cycle is repeated until the goal is reached. We denote such a RHC with the symbol $\mathcal{RH}(T, \delta)$. This process is schematized in Figure 2, where robot $\mathbf{R}_1$ plans during three cycles three trajectories that are tracked sequentially.

In other words, at time τ, we need to solve the following optimization problem:

$$\begin{aligned}
&\text{minimize} && \mathcal{J}(\mathbf{u}) = \int_\tau^{\tau+T} L(t, \mathbf{x}, \mathbf{u})\ \mathrm{d}t + V(\tau{+}T, \mathbf{x}) && (3)\\
&\text{where} && L(t, \mathbf{x}, \mathbf{u}) \text{ and} && (4)\\
& && V(t, \mathbf{x}) \text{ are defined by the user} && (5)\\
&\text{subject to} && \text{Equation 1} && (6)\\
& && \mathbf{u} \in \mathcal{U}, \mathbf{x} \in \mathcal{X} && (7)
\end{aligned}$$

where $\mathcal{U}$ and $\mathcal{X}$ represent user-defined admissible sets for the control inputs and state variables, respectively.

This optimization should generate an optimal policy $\mathbf{u}^\star$ in the time interval $[\tau, \tau + T]$:

$$\mathbf{u}^\star = \underset{u}{\operatorname{argmin}}\, \mathcal{J}(\mathbf{u}) \tag{8}$$

that achieves the rendezvous as time reaches infinity ($\tau \to \infty$) given a sampling time δ. According to Jadbabaie et al. [19], the convergence of the robots to a common rendezvous point is guaranteed for any δ and T such that $0 < \delta \leq T$ if and only if there exists a control $\bar{\mathbf{u}}$ with $\dot{V}(t, \mathbf{x}) + L(t, \mathbf{x}, \bar{\mathbf{u}}) \leq 0$ and $V(\cdot)$ is a control Lyapunov function. Hence, a good choice of both $L(\cdot)$ and $V(\cdot)$ is important and, thus, leaves the user with a difficult and constrained choice of the cost to minimize.

B. Remarks

Apart from the difficulty mentioned above, this optimization strategy presents three caveats: *(i)* it has a single point of failure, as its computation takes place on a single node (or robot), *(ii)* it assumes that communication is available (to transmit the observations) and synchronized, and *(iii)* the dimensionality of the problem increases linearly with the number of robots. Indeed, to solve this problem numerically, one needs to parametrize the input space given by $\mathbf{u}$ in terms of a finite number of coefficients, which have to be optimized. This can be achieved using N_b B-splines composed of N_c coefficients:

$$\mathbf{u}_i = \begin{bmatrix} u_i \\ \omega_i \end{bmatrix} = \begin{cases} \sum_{k=1}^{N_c} \begin{bmatrix} C_{u,i}^{1,k} \\ C_{\omega,i}^{1,k} \end{bmatrix} B_{k,p}(t) & \tau \leq t < t_1 \\ \quad \vdots & \\ \sum_{k=1}^{N_c} \begin{bmatrix} C_{u,i}^{N_b,k} \\ C_{\omega,i}^{N_b,k} \end{bmatrix} B_{k,p}(t) & t_{N_b-1} \leq t \leq t_{N_b} \end{cases} \tag{9}$$

where $B_{k,p}(t)$ is the B-spline basis function as defined in [7] with order p, and all t_i are a priori fixed constants (other parametrization such as piecewise-constant or piecewise-linear can also be used). Hence, there are $2N_bN_cN$ parameters to optimize and, as such, the computation time to solve the minimization problem increases exponentially with the number of robots.

III. Decentralized Receding Horizon Control

The main result of this paper is contained in this section where we address the aforementioned caveats, by proposing a decentralized RHC. We initially prove the stability and convergence of this decentralized RHC that achieves the rendezvous under any user-defined cost function. We then propose a computationally efficient approach to solve each optimization cycle using the differential flatness property of our system of robots.

A. Standard Policy

Definition 1 [Prediction function] At any time τ, each robot $\mathbf{R}_i$ is capable of predicting the future local coordinates x_{ij}, y_{ij} of each of its neighboring robot $\mathbf{R}_j$ using a *prediction function* f_{ij}

$$f_{ij}(\tau, t, \mathbf{u}_i, Z_i^{0..\tau}) = \begin{bmatrix} \hat{x}_{ij} \\ \hat{y}_{ij} \end{bmatrix} \tag{10}$$

where t is the time for which the prediction needs to be made, $\mathbf{u}_i$ is the control policy of robot $\mathbf{R}_i$ and $Z_i^{0..\tau} = \{Z_i(t) | t \in [0, \tau]\}$ is the list of all its past observations. The values $\hat{x}_{ij}$, $\hat{y}_{ij}$ are estimates of x_{ij}, y_{ij} at time t with

$$\begin{bmatrix} x_{ij} \\ y_{ij} \end{bmatrix} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{bmatrix} \begin{bmatrix} x_j - x_i \\ y_j - y_i \end{bmatrix} = \begin{bmatrix} \cos\alpha_{ij} \\ \sin\alpha_{ij} \end{bmatrix} e_{ij}.$$

Note that the prediction function f_{ij} can be implemented through Bayesian filtering. An example will be given in Section IV using an extended Kalman filter (EKF).

Definition 2 [Unbiased prediction function] An *unbiased* prediction function f_{ij} is defined such that, over all possible realizations of the observation list $Z_i^{0..\tau}$ and control inputs $\mathbf{u}_i$, its expected value at any time t is equal to the actual local coordinates of robot $\mathbf{R}_j$ with respect to $\mathbf{R}_i$, $\mathbb{E}_{\mathbf{u}_i, Z_i^{0..\tau}}[f_{ij}(\tau, t, \mathbf{u}_i, Z_i^{0..\tau})] = [x_{ij}, y_{ij}]^\mathsf{T}$. Additionally, we will denote by the symbol $\mathcal{F}^\mathcal{T}$ the set of all prediction functions f_{ij} such that f_{ij} is unbiased in the time set $\mathcal{T} \subseteq \mathbb{R}$:

$$\mathcal{F}^\mathcal{T} = \left\{ f_{ij} | \forall t \in \mathcal{T}, \mathbb{E}_{\mathbf{u}_i, Z_i^{0..\tau}}[f_{ij}(\tau, t, \mathbf{u}_i, Z_i^{0..\tau})] = \begin{bmatrix} x_{ij} \\ y_{ij} \end{bmatrix} \right\}$$

Theorem 1: Given a symmetric and connected group of N differential-wheeled robots $\mathbf{R}_1, \ldots, \mathbf{R}_N$ defined by the connectivity matrix $\mathbf{A}$, the receding horizon control $\mathcal{RH}(T_i, \delta_i)$, with $T_i > 0$ and $0 < \delta_i \leq T_i$, that solves the following optimization problem on each robot $\mathbf{R}_i$

$$\begin{aligned}
&\text{minimize} && \mathcal{J}_i(\mathbf{u}_i) = \int_\tau^{\tau+T} L_i(t, \hat{\mathbf{x}}_i, \mathbf{u}_i)\, \mathrm{d}t + V_i(\tau+T, \hat{\mathbf{x}}_i) && (11)\\
&\text{where} && \mathcal{J}_i(\mathbf{u}_i) \text{ is a user-defined functional} && (12)\\
& && \hat{\mathbf{x}}_i = \{[\hat{x}_{ij}, \hat{y}_{ij}]^\mathsf{T} | \mathbf{R}_j \in \mathcal{N}_i\} && (13)\\
& && f_{ij}(\tau, t, \mathbf{u}_i, Z_i^{0..\tau}) = [\hat{x}_{ij}, \hat{y}_{ij}]^\mathsf{T} && (14)\\
&\text{subject to} && \text{Equation 1} && (15)\\
& && \mathbf{u}_i \in \mathcal{U}_i, \mathbf{x}_i \in \mathcal{X}_i && (16)\\
& && f_{ij} \in \mathcal{F}^{[\tau, \tau+\delta]} && (17)\\
&\text{such that} && \exists k_{ij} = k_{ji} > 0 \text{ satisfying } u_i = \sum_{j=1}^{N} a_{ij} k_{ij} \hat{x}_{ij} && (18)\\
& && \exists t \geq \tau \text{ satisfying } \omega_i(t) \neq 0 && (19)
\end{aligned}$$

drives the group *almost surely* to a common rendezvous point.

Proof: Let us consider the stochastic Lyapunov candidate function:

$$\mathbb{V}(\mathbf{e}) = \sum_{i=1}^{N} \sum_{j=1}^{N} a_{ij} k_{ij} e_{ij}^2(t) \tag{20}$$

where $\mathbf{e} = [e_1, \ldots, e_{|\mathcal{E}|}]^\mathsf{T}$ with $e_k = e_{ij}$ for each edge $\mathsf{e}_k = (\mathbf{R}_i, \mathbf{R}_j) \in \mathcal{E}$ and k_{ij} a positive weight. It is clear that $\mathbb{V}(\mathbf{0}) = 0$ when all inter-neighbor distances e_{ij} are 0, for all $\mathbf{R}_i, \mathbf{R}_j \in \mathcal{N}_i$ and $\mathbb{V}(\mathbf{e}) > 0$ otherwise. Hence, according to [20], the system will *almost surely* converge to a common rendezvous point if and only if the expected sequence of Lyapunov values decreases everywhere but at the rendezvous point (i.e., the expected value of the derivative of $\mathbb{V}$ with respect to time is negative, $\mathbb{E}[\dot{\mathbb{V}}] < 0$ when $\mathbf{e} \neq 0$ and $\mathbb{E}[\dot{\mathbb{V}}] = 0$ when $\mathbf{e} = \mathbf{0}$), and the network of robots is connected.

By realizing that $e_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$, we have $\dot{e}_{ij} = -u_i \cos\alpha_{ij} - u_j \cos\alpha_{ji}$ (using Equation 1). Thus,

$$\dot{\mathbb{V}}(\cdot) = 2\sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij}k_{ij}e_{ij}\dot{e}_{ij} \tag{21}$$

$$= -2\sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij}u_i k_{ij}e_{ij}\cos\alpha_{ij} - 2\sum_{i=1}^{N}\sum_{j=1}^{N} a_{ji}u_i k_{ji}e_{ji}\cos\alpha_{ij} \tag{22}$$

and since $\mathbf{A}$ is symmetric ($a_{ij} = a_{ji}$ and $k_{ij} = k_{ji}$) and $e_{ij} = e_{ji}$, we obtain

$$\dot{\mathbb{V}}(\cdot) = -4\sum_{i=1}^{N} u_i \sum_{j=1}^{N} a_{ij}k_{ij}e_{ij}\cos\alpha_{ij} = -4\sum_{i=1}^{N} u_i \sum_{j=1}^{N} a_{ij}k_{ij}x_{ij} \tag{23}$$

which yields

$$\begin{aligned}
\mathbb{E}[\dot{\mathbb{V}}(\cdot)] &= -4\sum_{i=1}^{N}\mathbb{E}[u_i]\sum_{j=1}^{N} a_{ij}k_{ij}x_{ij} \\
&\overset{\text{Eq. 18}}{=} -4\sum_{i=1}^{N}\left(\sum_{j=1}^{N} a_{ij}k_{ij}\mathbb{E}[\hat{x}_{ij}]\right)\left(\sum_{j=1}^{N} a_{ij}k_{ij}x_{ij}\right) \\
&\overset{\text{Eq. 35}}{=} -4\sum_{i=1}^{N}\left(\sum_{j=1}^{N} a_{ij}k_{ij}x_{ij}\right)^2 \\
&\leq 0.
\end{aligned} \tag{24}$$

We notice that $\mathbb{E}[\dot{\mathbb{V}}](\mathbf{0}) = 0$, but also that $\mathbb{E}[\dot{\mathbb{V}}](\mathbf{e})$ may be zero when $\mathbf{e} \neq \mathbf{0}$. However, the set $\mathcal{S} = \{\mathbf{e} | \mathbb{E}[\dot{\mathbb{V}}](\mathbf{e}) = 0\}$ does not contain any trajectories of the system, except the trivial trajectory $\mathbf{e}(t) = \mathbf{0}$, since Equation 19 guarantees that $\omega_i(t) \neq 0$. Hence, according to the Krasovskii-LaSalle principle, the continuous-time sequence of Lyapunov values is a supermatingale and, thus, the system *almost surely* converges to a common rendezvous point. Note that the same proof hold when $k_{ij}(t)$ is a positive scalar function ($k_{ij}(t) > 0, \forall t \in \mathbb{R}$). ■

At the cost of adding a constraint, and if we parametrize u_i and ω_i using N_b splines of N_c coefficients (as shown in Equation 9), we now need to optimize only $2N_bN_c$ coefficients locally on every robot, rather than $2N_bN_cN$ on a single robot. Still, a direct implementation of this RHC requires the numerical integration of the kinematic differential Equation 1. This extra computation can slow down the optimization by several orders of magnitude, which can induce both delays as well as the violation of the optimality of the prediction function (as the time range over which it needs to be optimal gets larger). Hence, like in the NTG software [29], one should map these dynamic constraints to algebraic ones if the system is flat to ensure that the sampling time δ is as small as possible (see [31] for a comparison of the efficiency of this approach). This approach relaxes the requirements on the prediction function and guarantees, in practice, the convergence of the group of robots.

B. Differential Flatness

In this section, we show that the differential-wheeled robot with three inputs is a *differentially flat* system [17] with x_i and y_i as flat outputs. Indeed, from Equation 1, assuming that robot $\mathbf{R}_i$ moves forward, we can see that

$$\frac{\dot{x}_i}{\dot{y}_i} = \frac{\cos\theta_i}{\sin\theta_i} \Rightarrow \theta_i = \text{atan2}(\dot{y}_i, \dot{x}_i) \tag{25}$$

and that

$$\dot{x}_i = u_i\cos\theta_i \Rightarrow u_i = \frac{\dot{x}_i}{\cos(\text{atan2}(\dot{y}_i, \dot{x}_i))} \tag{26}$$

$$\dot{\theta}_i = \omega_i \Rightarrow \omega_i = \frac{\dot{x}_i\ddot{y}_i + \ddot{x}_i\dot{y}_i}{\dot{x}_i^2 + \dot{y}_i^2}. \tag{27}$$

Equation 26 can also be solved with $\dot{y}_i$ when $\cos\theta_i = 0$. Hence, all the state variables x_i, y_i and θ_i and the control inputs u_i and ω_i can be expressed by the trajectory of the robot $\mathbf{R}_i$ (given by x_i and y_i) and its derivatives if the direction of motion is known. Note that if the robot moves backward, we have $\theta_i = \text{atan2}(\dot{y}_i, \dot{x}_i) + \pi$ instead.

C. Efficient Policy

Finally, to avoid the numerical integration of Equation 1, and since the system is flat, it is preferable that each robot $\mathbf{R}_i$ directly optimizes its own trajectory:

$$\text{minimize } \mathcal{J}_i(\bar{\mathbf{x}}_i, \dot{\bar{\mathbf{x}}}_i, \ddot{\bar{\mathbf{x}}}_i) = \int_{\tau}^{\tau+T} L_i(t, \hat{\mathbf{x}}_i, \cdot)\mathrm{d}t + V_i(\tau+T, \hat{\mathbf{x}}_i) \tag{28}$$

$$\text{where } \mathcal{J}_i(\cdot) \text{ is a user-defined functional} \tag{29}$$

$$\hat{\mathbf{x}}_i = \{[\hat{x}_{ij}, \hat{y}_{ij}]^\mathsf{T} | \mathbf{R}_j \in \mathcal{N}_i\} \tag{30}$$

$$f_{ij}(\tau, t, \mathbf{u}_i, Z_i^{0..\tau}) = [\hat{x}_{ij}, \hat{y}_{ij}]^\mathsf{T} \tag{31}$$

$$\text{subject to } \mathbf{u}_i \in \mathcal{U}_i, \mathbf{x}_i \in \mathcal{X}_i \tag{32}$$

$$f_{ij} \in \mathcal{F}^{[\tau, \tau+\delta]} \tag{33}$$

$$\text{such that } \exists k_{ij} = k_{ji} > 0 \text{ satisfying } u_i = \sum_{j=1}^{N} a_{ij}k_{ij}\hat{x}_{ij} \tag{34}$$

$$f_{ij} \in \mathcal{F}^{[\tau, \tau+\delta]} \tag{35}$$

$$\exists t \geq \tau \text{ satisfying } \omega_i(t) \neq 0 \tag{36}$$

where $\mathbf{u}_i$ is given by Equations 26 and 27 and depends only on $\bar{\mathbf{x}}_i = [x_i, y_i]^\mathsf{T}$ and its derivatives.

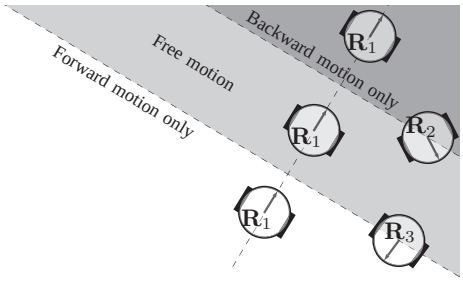


Fig. 3. This is a geometrical interpretation of the constraint of Equation 18. Robot $\mathbf{R}_1$ needs to move forward if it is below the projection of $\mathbf{R}_3$ on its current x-axis, that is x_{1j} is positive for all robots $\mathbf{R}_j \in \mathcal{N}_1$, it needs to move backward if the scalar product is negative and is free to move as it likes otherwise.

Using B-splines to represent x_i and y_i enables easy computation of the derivatives and ensures the continuity of the trajectory. The direction of motion of $\mathbf{R}_i$ is then selected based on Equation 34 for each B-spline. Finally, we note that this flatness parametrization reduces the computation time needed to solve our minimization problem by a factor of about a hundred (as hinted by [31]).

D. Remarks

For completeness, we make a few remarks on Theorem 1 that highlight its usage and limitations.

Remark 1: The condition given by Equation 35 requires that the prediction filter is optimal in the sense that the estimation of the neighboring robot positions is unbiased with respect to the observation and motion noise.

Remark 2: Equation 18 might be hard to interpret as the weight k_{ij} may vary in time. In practice and although not strictly equivalent, we can transform it into the geometrical constraints shown in Figure 3. Given a fixed orientation for a robot $\mathbf{R}_i$, one can separate the state space in three regions: a forward motion region, a free region, and a backward motion region. The boundaries between these regions are given by the neighboring robots yielding the projection on the x-axis of robot $\mathbf{R}_i$ that are the farthest away from each other. This condition is equivalent to

$$\begin{cases} u_i > 0 \;\text{ if }\; \min_{\mathbf{R}_j \in \mathcal{N}_i}(\hat{x}_{ij}) > 0 \\ u_i < 0 \;\text{ if }\; \max_{\mathbf{R}_j \in \mathcal{N}_i}(\hat{x}_{ij}) < 0 \\ u_i \in \mathbb{R} \;\text{ otherwise} \end{cases} \tag{37}$$

Remark 3: The choice of $\mathcal{U}_i$ and $\mathcal{X}_i$ should not violate Equation 18. In other words, there should always be a solution to the minimization problem.

Remark 4: Both Equations 35 and 18 may be replaced by a more general and less restrictive condition which only requires the expected value of the forward motion u_i to be of the same sign as its equivalent noise-free control law:

$$\exists k_{ij} > 0 \text{ s.t. } \text{sign}(\mathbb{E}_{\mathbf{u}_i, Z_i^{0\cdots\tau}}[u_i]) = \text{sign}(\sum_{j=1}^{N} a_{ij} k_{ij} x_{ij}).$$

However, this condition is not practical and cannot be directly used, as the true positions x_{ij} of the neighboring robots are unknown.

Remark 5: The last condition given by Equation 19 is a mild condition expressing that a robot should eventually have a non-zero rotational motion. It ensures that the rendezvous can happen by avoiding the degenerative case where robots are simply moving forward and backward. In practice, however, due to the observation noise, any control law that may take non-zero rotational motion for a given set of local coordinates will eventually achieve the rendezvous and, thus, this condition can be safely ignored.

Remark 6: Theorem 1 implicitly states that both the time horizon T_i and the sampling time δ_i may be different amongst robots. Hence, there is no direct need to synchronize the robots. This control strategy is fundamentally different from any distributed RHC approach where the original RHC computation would be distributed across robots and would require a tight synchronization [11]. Our decentralized approach, although less optimal than its centralized version, allows for a robust and totally asynchronous control, and, as we will see in Section V, achieves good performance and guarantees the rendezvous.

Remark 7: Finally, it is worth mentioning that *(i)* if the prediction is perfect ($f_{ij}(\cdot) = [x_{ij}, y_{ij}]^\mathsf{T}$), this minimization problem is equivalent to its centralized counterpart as $\mathcal{J} = \sum_i^N \mathcal{J}_i(\cdot)$, and *(ii)* if the time horizon T approaches zero ($T \to 0$, $0 < \delta \leq T$), Theorem 1 degenerates into the third theorem of [18] which does not allow trajectory optimization.

IV. Implementation on Real Robots

In this section, we address the implementation of our RHC on real robots. In particular, we will detail our strategy to account for delays induced by our real-time optimization and the implementation of our prediction function.

A. Computational Delays

In RHC, the optimized trajectory is followed during a time δ during which no feedback from the environment is observed. After δ seconds, feedback from the environment is incorporated to re-optimize the trajectory. In practice, the amount of time δ dedicated to follow the trajectory is not fixed. Indeed, one often prefers to optimize the trajectory as fast as possible and use the result as early as possible. The sampling time δ then directly relates to the computation time needed to optimize the new trajectory. It is clear that while the optimization takes place, the robot continues to move according to the old trajectory which may result in a mismatch between the optimized position and the current position at the time when the optimization is completed. Hence, the robot $\mathbf{R}_i$ needs to reacquire (and track) the optimized trajectory. To do so, it needs to know its current position with respect to the desired new position, hereafter denoted by the coordinates (x_d, y_d). This can easily be achieved by integrating the open-loop controls or, for more precision, by using odometry measurements (in our case given by wheel encoders which are deployed on most differential-wheeled robots). Figure 4 shows a robot with its desired trajectory. According to the desired trajectory, robot

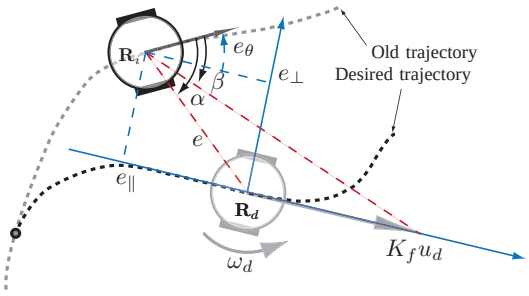


Fig. 4. Schema of the quantities used by the control law in Equation 38 that enable $\mathbf{R}_i$ to reach a trajectory given by the virtual robot $\mathbf{R}_d$ (desired trajectory) after having followed the old trajectory for *too long*.

$\mathbf{R}_i$ should be located at the position indicated by the virtual robot reference $\mathbf{R}_d$. $\mathbf{R}_i$ is able to calculate the range e and the bearing α to $\mathbf{R}_d$. Using the equations in Section III-B, the robot can identify the orientation $-e_\theta$ (with respect to itself), the forward motion u_d and rotational motion ω_d of $\mathbf{R}_d$. Note that β is the bearing to the point located at a distance $K_f u_d$ in front of $\mathbf{R}_d$. We propose the following control law when $\mathbf{R}_d$ moves forward:

$$\begin{cases} u_i &= K_u e \cos\alpha + u_d \\ \omega_i &= K_\omega e \sin\alpha + K_b \beta + \omega_d \end{cases} \tag{38}$$

with K_u, K_ω, K_b and K_f all positive constants. An equivalent control law can be found when $\mathbf{R}_d$ moves backward. Although omitted here for conciseness, it can be shown that this control law is stable and converges to the desired trajectory.

This strategy bears resemblance to the third strategy proposed by Milam et al. [30] to account for computation delays, with the exception that, instead of blindly applying the optimized control inputs, we compute corrected control inputs based on the optimized trajectory using a tracking layer.

B. Prediction Function

In this section, we detail our prediction function which takes the form of an EKF. It takes the measurements and control actions as input and yields an estimate of the position of neighboring robots during the finite time horizon T.

1) Measurements: A robot $\mathbf{R}_i$ will track its neighbors' positions and speeds in its local coordinate frame. The local state of robot $\mathbf{R}_j$ with respect to $\mathbf{R}_i$ is then $\mathbf{x}_{ij} = [x_{ij}, y_{ij}, \theta_{ij}, u_j, \omega_j]^\mathsf{T}$ where $\theta_{ij} = \theta_j - \theta_i$. In our case study, three possible types of measurements are considered.

As stated in Section I-A, we have range and bearing measurements given by:

$$h_i^{(1)}(\mathbf{x}_{ij}) = z_{ij} = \begin{bmatrix} e_{ij} \\ \alpha_{ij} \end{bmatrix} = \begin{bmatrix} \sqrt{x_{ij}^2 + y_{ij}^2} \\ \text{atan2}(y_{ij}, x_{ij}) \end{bmatrix} + \epsilon_1 \tag{39}$$

where $\epsilon_1 = \epsilon_z$ is distributed according to a multivariate zero-mean normal distribution with covariance R_1: $\epsilon_1 \sim \mathcal{N}(0, R_1)$. A measurement campaign performed in [18] confirms that the noise ϵ_1 is normally distributed with $R_1 \approx [0.0221 \;\; -0.0011; \;\; -0.0011 \;\; 0.0196]$.

Additionally, neighboring robots may share their observations from time to time, thus

$$h_i^{(2)}(\mathbf{x}_{ij}) = \begin{bmatrix} e_{ji} \\ \alpha_{ji} \end{bmatrix} = \begin{bmatrix} \sqrt{x_{ij}^2 + y_{ij}^2} \\ \text{atan2}(y_{ij}, x_{ij}) - \theta_{ij} + \pi \end{bmatrix} + \epsilon_2 \tag{40}$$

with $\epsilon_2 \sim \mathcal{N}(0, R_2)$, and their forward and rotational control inputs,

$$h_i^{(3)}(\mathbf{x}_{ij}) = \begin{bmatrix} u_j \\ \omega_j \end{bmatrix} + \epsilon_3 \tag{41}$$

with $\epsilon_3 \sim \mathcal{N}(0, R_3)$, using a wireless communication channel.

2) Kalman Filtering: We can combine the above observation models with the motion of robots $\mathbf{R}_i$ and $\mathbf{R}_j$, given by Equation 1, into an EKF. Hence, given all previous observations $Z_i^{0..\tau}$ until time τ and by discretizing Equation 1 (i.e., using Euler's integration), the prediction function of the local state of $\mathbf{R}_j$ made at time τ for time $t + \Delta t$ becomes

$$f_{ij}(\tau, t + \Delta t, \mathbf{u}_i, Z_i^{0..\tau}) = [\hat{x}_{ij}(t + \Delta t), \hat{y}_{ij}(t + \Delta t)]^\mathsf{T}$$

with

$$\hat{f}_{ij}(t{+}\Delta t, \hat{\mathbf{x}}_{ij}) = \begin{cases} \hat{x}_j(t + \Delta t) &= \Delta x \cos d\theta + \Delta y \sin \Delta\theta \\ \hat{y}_j(t + \Delta t) &= \Delta y \cos d\theta - \Delta x \sin \Delta\theta \\ \hat{\theta}_j(t + \Delta t) &= \hat{\theta}_j(t) + \hat{\omega}_j(t)\Delta t - \Delta\theta \\ \hat{u}_j(t + \Delta t) &= \hat{u}_j(t) \\ \hat{\omega}_j(t + \Delta t) &= \hat{\omega}_j(t) \end{cases} \tag{42}$$

where $\Delta x = \hat{x}_j(t) + \hat{u}_j(t)\Delta t \cos(\hat{\theta}_j(t)) - u_i(t)\Delta t$, $\Delta y = \hat{y}_j(t) + \hat{u}_j(t)\Delta t \sin(\hat{\theta}_j(t))$, $\Delta\theta = \omega_i(t)\Delta t$ and Δt is a constant time-step duration. For completeness, we state the *predicted estimate covariance*:

$$P_{ij}(t{+}\Delta t, \hat{\mathbf{x}}_{ij}) = F_{ij}(t, \hat{\mathbf{x}}_{ij}) P_{ij}(t, \hat{\mathbf{x}}_{ij}) F_{ij}^\mathsf{T}(t, \hat{\mathbf{x}}_{ij}) + Q \tag{43}$$

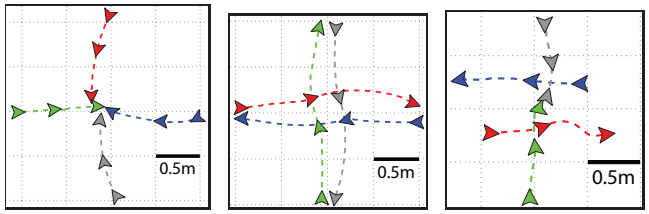
where $P_{ij}(t, \hat{\mathbf{x}}_{ij})$ is the covariance of the state estimate, $F_{ij}(t, \hat{\mathbf{x}}_{ij})$ is the Jacobian matrix of $\hat{f}_{ij}(t, \hat{\mathbf{x}}_{ij})$ and Q is the covariance of the process noise. We can now compose in succession Equations 42 and 43 for several time-steps and get a prediction of the future positions of neighboring robots as well as their associated uncertainties. For simplicity, we have purposefully omitted the *update* equations of the EKF (that correct the estimate $\hat{\mathbf{x}}_j$ when a new measurement is made) and redirect the reader towards [25] for more information.

V. Experiments

In this section, we validate our approach by comparing it with a reactive control law capable of achieving the rendezvous. Simultaneously, we discuss the performance of our system by analyzing the minimization of a specific user-defined cost function.

A. Setup

Experiments were performed using Khepera III robots [32]. This robot has a diameter of 12 cm, making it appropriate for multi-robot indoor experiments. As shown in Figure 1(a), we equip each robot with a range and bearing module allowing for inter-robot positioning. The robots



(a) (b) (c)

Fig. 5. Two runs performed on real robots using the predictive controller and tracked using SwisTrack. (a) The robots achieve the rendezvous — Scenario (a). (b) The robots cross each other and rendezvous with their desired goal positions — Scenario (c). (c) The two robots in the vertical axis rendezvous whilst the other two cross the arena — Scenario (d).

are placed in a $3\times3\text{m}^2$ arena and we perform four sets of experiments:

Scenario (a) Four robots are randomly placed in the arena and form a complete graph (all robots are neighbors). Their task is to perform the rendezvous. An example of trajectories obtained by the real robots on this scenario is shown in Figure 5(a).

Scenario (b) Two robots $\mathbf{R}_1$ and $\mathbf{R}_2$ are placed 2 meters apart, facing each other. Each robot has to reach the initial location of the other robot. These locations can be formalized by 2 additional motion-less robots $\mathbf{R}_3$ and $\mathbf{R}_4$ (whose relative positions are artificially fed to the robots). More formally, we have $\mathbf{x}_1(0) \approx [0.5, 1.53, 0]^\mathsf{T}$, $\mathbf{x}_2(0) \approx [2.5, 1.47, \pi]^\mathsf{T}$, $\mathbf{x}_3(0) = [2.5, 1.53, 0]^\mathsf{T}$, $\mathbf{x}_4(0) = [0.5, 1.47, 0]^\mathsf{T}$, $u_3 = u_4 = \omega_3 = \omega_4 = 0$ and $\mathcal{N}_1 = \{\mathbf{R}_3\}$, $\mathcal{N}_2 = \{\mathbf{R}_4\}$. This scenario not only tests how a user-defined cost (explained later) can cope with obstacle avoidance, but also how each robot is able to rendezvous with a fixed goal position (given here by robots $\mathbf{R}_3$ and $\mathbf{R}_4$).

Scenario (c) Like the previous scenario but with four robots. This is a complex crossing and is an effective test-bed for analyzing the ability to optimize the trajectories quickly. An example of trajectories obtained by the real robots on this scenario is shown on Figure 5(b).

Scenario (d) This scenario involves two robots having to rendezvous and two other robots disturbing this rendezvous maneuver by crossing the arena. An example of trajectories obtained by the real robots on this scenario is shown in Figure 5(c).

The ground truth position and orientation of each robot is monitored with SwisTrack [24], an open-source tracking software. For each set of experiments, we do ten runs with both the **reactive** controller proposed in [18] (on top of which we add an obstacle avoidance control as explained in [13]) and the **RHC** controller of Section III-C. Constants of both controllers are tuned such that the average forward motion is about 12 cm/s (about one robot size per second).

B. User-defined Cost and Constraints

For the purpose of our experimental setup and as an example, we design a local cost function for each robot $\mathbf{R}_i$ that accounts for the energy spent in actuation and the risk of collision with robots that are not connected (in the sense of the underlying graph defined by $\mathbf{A}$). For brevity, we consider that a robot $\mathbf{R}_i$ is also able to observe and predict the position of non-connected robots $\mathbf{R}_j$ in its vicinity and denote the set of all such robots $\mathbf{R}_j$ by $\mathcal{O}_i$. Referring to Equation 28, we set

$$L_i(\cdot) = L_i^{(1)}(\cdot) + \sum_{\mathbf{R}_j \in \mathcal{O}i} L_{ij}^{(2)}(\cdot) \tag{44}$$

$$L_i^{(1)}(\cdot) = k_1 u_i^2 + k_2 \omega_i^2 \tag{45}$$

$$L_{ij}^{(2)}(\cdot) = k_3 \iint_{[x,y]^\mathsf{T} \in \mathcal{D}} \Phi\left([x,y]^\mathsf{T} | f_{ij}(t,\cdot), P_{ij}(t,\cdot)\right) \mathrm{d}x\mathrm{d}y \tag{46}$$

$$V_i(\cdot) = k_4 \sum_{\mathbf{R}_j \in \mathcal{N}_i} e_{ij}(T)^2 \tag{47}$$

where $\Phi(\cdot|\mu, \Sigma)$ is the multivariate Gaussian probability density function with mean μ and covariance Σ, $\mathcal{D}$ is the Euclidean disk of radius $2R$ centered at zero and R is the radius of a robot. We set $k_1 = 0.1$, $k_2 = 3$, $k_3 = 1/|\mathcal{O}_i|$ and $k_4 = 2$.[1]

Additionally, we add a constraint on the maximal wheel speed ($\omega_{\max} = 3$ rad/s):

$$\mathcal{U}_i = \{\mathbf{u}_i | u_i \in [l|\omega_i|/2 - r\omega_{\max}, -l|\omega_i|/2 + r\omega_{\max}]\},$$

where l is the axle length. We pose no constraints on the state space (i.e., $\mathcal{X}_i = \emptyset$) and set the time horizon T to 3 seconds.[2]

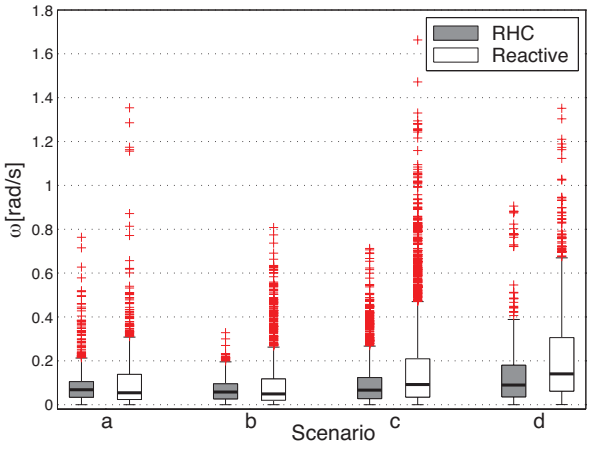


Fig. 6. Boxplot of the absolute rotational motion of all robots for all scenarios. The scenario complexity increases from left to right. The red crosses represent outliers. We can clearly see that the RHC outperforms the reactive controller both in terms of performance (minimizing the rotational speed) and robustness (lower variability).

C. Results

The absolute rotational speed of all robots observed every tenth of a second is reported in Figure 6 for all scenarios

[1] A poor choice of the cost function may result in *apparent* deadlock situations (i.e., robots make very slow progress towards the rendezvous). In such an eventuality, one can detect the deadlock by observing the forward motion and solve it by relaxing the user-defined constraints (soft or hard).

[2] Since u_i may take both positive and negative values and there are no state space constraints, this choice of $\mathcal{U}_i$, $\mathcal{X}_i$ respects Remark 3.

and controllers in the form of boxplots. In our case, the rotational speed is a valid performance indicator, since the forward speed was the same across controllers and scenarios (i.e., u_i is constant), all runs were collision-free (i.e., $L_{ij}^{(2)}(\cdot) \ll L_i^{(1)}(\cdot)$) and all rendezvous maneuvers succeeded (i.e., $V_i(\cdot) \to 0$).

We can observe that, compared to the traditional reactive controller, the rotational motion of the RHC is generally smaller, hence the cost function is minimized. This result demonstrates the feasibility and the validity of our approach under a wide range of test scenarios given a specific user-defined cost. Additionally, the lower variability of the rotational speed confirms the robustness of the RHC.

Finally, our approach allows to generate trajectories that are meaningful with respect to a user-defined cost function. It can account for the motion of other robots (or elements detectable with any on-board sensors) in the environment and plan accordingly whilst respecting mathematical guarantees on the achievement of the rendezvous.

VI. Conclusion

We presented a decentralized receding horizon control capable of generating, in real-time, optimal rendezvous trajectories for differential-wheeled robots under actuation constraints and noisy sensing. In particular, we guarantee that robots will meet at a common location, independently of the user-defined optimality criterion. All properties were successfully demonstrated on a real hardware platform, namely the Khepera III robot.

References

[1] H. Ando, Y. Oasa, I. Suzuki, and M. Yamashita. Distributed memoryless point convergence algorithm for mobile robots with limited visibility. *IEEE Transactions on Robotics and Automation*, 15(5):818–828, 1999.

[2] R. Beard, T. McLain, D. Nelson, D. Kingston, and D. Johanson. Decentralized cooperative aerial surveillance using fixed-wing miniature UAVs. *Proceedings of the IEEE*, 94(7):1306–1324, 2006.

[3] C. Caicedo-Nunez and M. Zefran. Probabilistic guarantees for rendezvous under noisy measurements. In *American Control Conference, 2009*, pages 5180–5185, 2009.

[4] M. Cao, A. Morse, and B. Anderson. Reaching a consensus in a dynamically changing environment: A graphical approach. *SIAM Journal on Control and Optimization*, 47(2):575–601, 2008.

[5] T. Carter and M. Humi. Fuel-optimal rendezvous near a point in general keplerian orbit. *Journal of Guidance, Control, and Dynamics*, 10:567–573, 1987.

[6] D. H. Chyung. Time optimal rendezvous of three linear systems. *Journal of Optimization Theory and Applications*, 12:242–247, 1973.

[7] C. de Boor, editor. *A Practical Guide to Splines*. Springer, 1978. ISBN 978-0-387-95366-3.

[8] D. Dimarogonas and K. Johansson. Further results on the stability of distance-based multi-robot formations. In *American Control Conference*, pages 2972–2977, 2009.

[9] D. Dimarogonas and K. Kyriakopoulos. On the rendezvous problem for multiple nonholonomic agents. *IEEE Transactions on Automatic Control*, 52(5):916–922, 2007.

[10] D. V. Dimarogonas, S. G. Loizou, K. J. Kyriakopoulos, and M. M. Zavlanos. A feedback stabilization and collision avoidance scheme for multiple independent non-point agents. *Automatica*, 42(2):229–243, 2006.

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

[12] E. Eisenberg and D. Gale. Consensus of subjective probabilities: The pari-mutuel method. *The Annals of Mathematical Statistics*, 30(1): 165–168, 1959.

[13] R. Falconi, L. Sabattini, C. Secchi, C. Fantuzzi, and C. Melchiorri. A graph–based collision–free distributed formation control strategy. In *18th IFAC World Congress*, 2011. doi: 10.3182/20110828-6-IT-1002.02450.

[14] J. Fax and R. Murray. Information flow and cooperative control of vehicle formations. *IEEE Transactions on Automatic Control*, 49(9): 1465–1476, 2004.

[15] W. Fehse, editor. *Automated Rendezvous and Docking of Spacecraft*. Cambridge University Press, 2003. ISBN 978-0-521-82492-7.

[16] F. Findeisen and F. Allgöwer. An introduction to nonlinear model predictive control. *Benelux Meeting on Systems and Control*, pages 119–141, 2002.

[17] M. Fliess, J. Levine, P. Martin, and P. Rouchon. On differentially flat nonlinear system. In *NOLCOS*, pages 408–412, 1992.

[18] S. Gowal and A. Martinoli. Bayesian rendezvous for distributed robotic systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2765–2771, 2011.

[19] A. Jadbabaie, J. Yu, and J. Hauser. Unconstrained receding-horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, 46(5):776–783, 2001.

[20] H. J. Kushner. *Stochastic stability and control*. Academic Press New York, 1967. ISBN 0124301509.

[21] J. Lawton, R. Beard, and B. Young. A decentralized approach to formation maneuvers. *IEEE Transactions on Robotics and Automation*, 19(6):933–941, 2003.

[22] J. Lin, A. Morse, and B. Anderson. The multi-agent rendezvous problem. In *Proceedings of the IEEE Conference on Decision and Control*, volume 2, pages 1508–1513, 2003.

[23] J. Lin, A. Morse, and B. Anderson. The multi-agent rendezvous problem - the asynchronous case. In *Proceedings of the IEEE Conference on Decision and Control*, volume 2, pages 1926–1931, 2004.

[24] T. Lochmatter, P. Roduit, C. Cianci, N. Correll, J. Jacot, and A. Martinoli. Swistrack - a flexible open source tracking software for multi-agent systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4004–4010, 2008.

[25] P. S. Maybeck. *Stochastic models, estimation, and control*, volume 141 of *Mathematics in Science and Engineering*. Academic Press, 1979. ISBN 978-0-124-80701-3.

[26] T. McLain, P. Chandler, S. Rasmussen, and M. Pachter. Cooperative control of UAV rendezvous. In *American Control Conference, 2001. Proceedings of the 2001*, volume 3, pages 2309–2314, 2001.

[27] P. Meschler. Time-optimal rendezvous strategies. *IEEE Transactions on Automatic Control*, 8(4):279–283, 1963.

[28] A. Miele, M. Weeks, and M. Ciarcià. Optimal trajectories for spacecraft rendezvous. *Journal of Optimization Theory and Applications*, 132:353–376, 2007.

[29] M. Milam, K. Mushambi, and R. Murray. A new computational approach to real-time trajectory generation for constrained mechanical systems. In *IEEE Conference on Decision and Control*, volume 1, pages 845–851, 2000.

[30] M. Milam, R. Franz, J. Hauser, and R. Murray. Receding horizon control of vectored thrust flight experiment. *IEE Proceedings on Control Theory and Applications*, 152(3):340–348, 2005.

[31] N. Petit, M. B. Milam, and R. M. Murray. Inversion based constrained trajectory optimization. In *5th IFAC symposium on nonlinear control systems*, 2001.

[32] A. Prorok, A. Arfire, A. Bahr, J. Farserotu, and A. Martinoli. Indoor navigation research with the Khepera III mobile robot: An experimental baseline with a case-study on ultra-wideband positioning. In *2010 International Conference on Indoor Positioning and Indoor Navigation*, 2010. doi: 10.1109/IPIN.2010.5647880.

[33] W. Ren. Distributed attitude consensus among multiple networked spacecraft. In *American Control Conference*, 2006. doi: 10.1109/ACC.2006.1656474.

[34] W. Ren and R. Beard. *Distributed Consensus in Multi-vehicle Cooperative Control: Theory and Applications*. Springer Publishing Company, Incorporated, 2007. ISBN 1848000146, 9781848000148.

[35] C. Reynolds. Flocks, herds and schools: A distributed behavioral model. *SIGGRAPH Computer Graphics*, 21(4):25–34, 1987.