Flocking algorithm for autonomous flying robots

Bioinspir. Biomim. **9** (2014) 025012 (11pp) doi:10.1088/1748-3182/9/2/025012

Flocking algorithm for autonomous flying robots

Csaba Virágh[1], **Gábor Vásárhelyi**[1,2], **Norbert Tarcai**[1], **Tamás Szörényi**[1], **Gergő Somorjai**[1,2], **Tamás Nepusz**[1,2] **and Tamás Vicsek**[1,2]

[1] ELTE Department of Biological Physics, 1117 Budapest, Pázmány Péter Sétány 1/A, Hungary
[2] MTA-ELTE Statistical and Biological Physics Research Group, 1117 Budapest, Pázmány Péter Sétány 1/A, Hungary

E-mail: viraghcs@hal.elte.hu

Received 20 February 2014, revised 22 April 2014
Accepted for publication 23 April 2014
Published 22 May 2014

Abstract
Animal swarms displaying a variety of typical flocking patterns would not exist without the underlying safe, optimal and stable dynamics of the individuals. The emergence of these universal patterns can be efficiently reconstructed with agent-based models. If we want to reproduce these patterns with artificial systems, such as autonomous aerial robots, agent-based models can also be used in their control algorithms. However, finding the proper algorithms and thus understanding the essential characteristics of the emergent collective behaviour requires thorough and realistic modeling of the robot and also the environment. In this paper, we first present an abstract mathematical model of an autonomous flying robot. The model takes into account several realistic features, such as time delay and locality of communication, inaccuracy of the on-board sensors and inertial effects. We present two decentralized control algorithms. One is based on a simple self-propelled flocking model of animal collective motion, the other is a collective target tracking algorithm. Both algorithms contain a viscous friction-like term, which aligns the velocities of neighbouring agents parallel to each other. We show that this term can be essential for reducing the inherent instabilities of such a noisy and delayed realistic system. We discuss simulation results on the stability of the control algorithms, and perform real experiments to show the applicability of the algorithms on a group of autonomous quadcopters. In our case, bio-inspiration works in two ways. On the one hand, the whole idea of trying to build and control a swarm of robots comes from the observation that birds tend to flock to optimize their behaviour as a group. On the other hand, by using a realistic simulation framework and studying the group behaviour of autonomous robots we can learn about the major factors influencing the flight of bird flocks.

S Online supplementary data available from stacks.iop.org/BB/9/025012/mmedia

Keywords: swarm robotics, flying robot flock, collective motion, distributed control, autonomous navigation

(Some figures may appear in colour only in the online journal)

1. Introduction

Collective motion is an impressive phenomenon, that can be observed in a wide range of biological systems, such as fish schools, bird flocks, herds of mammals or migrating cells [1]. These systems produce the same universal feature: the velocity vectors of neighbouring individuals tend to become parallel to each other. This behaviour and the underlying control mechanism seem to be a prerequisite of safe, stable and collision-free motion. Therefore, it might be advantageous to incorporate the mathematical models that reproduce group flight patterns into the control of artificial systems, a group of autonomous flying robots, for example. By *autonomous* we mean that every agent uses on-board sensors to measure its state and performs all controlling calculations with an on-board computer, i.e., the control system is decentralized. This definition prohibits central processing of the group dynamics by an external computer, but allows the use of, e.g., on-board GPS devices for external reference of position. Our study is valid for any kind of object that is capable of moving in an arbitrary direction independently of its orientation, within a reasonable velocity range (including zero velocity hovering). Typical flying robots that satisfy this criteria are the so-called quadro-, hexa-, and octocopters, commonly named as multicopters.

According to Reynolds, collective motion of various kinds of entities can be interpreted as a consequence of three simple principles [2]: repulsion in short range to avoid collisions, a local interaction called *alignment rule* to align the velocity vectors of nearby units and preferably global positioning constraint to keep the flock together. These rules can be interpreted in mathematical form as an agent-based model, i.e., a (discrete or continuous) dynamical system that describes the time-evolution of the velocity of each unit individually.

The simplest agent-based models of flocking describe the alignment rule as an explicit mathematical axiom: every unit aligns its velocity vector towards the average velocity vector of the units in its neighbourhood (including itself) [3]. It is possible to generalize this term by adding coupling of accelerations [4], preferred directions [5] and adaptive decision-making schemes to extend the stability for higher velocities [6]. In other (more specific) models, the alignment rule is a consequence of interaction forces [7] or velocity terms based on over-damped dynamics [8].

An important feature of the alignment rule terms in flocking models is their locality; units align their velocity towards the average velocity of other units within a limited range only. In flocks of autonomous robots, the communication between the robots usually has a finite range. In other words, the units can send messages (e.g., their positions and velocities) only to other nearby units. Another analogy between nature based flocking models and autonomous robotic systems is that both can be considered to be based on agents, i.e., autonomous units subject to some system-specific rules. In flocking models, the velocity vectors of the agents evolve individually through a dynamical system. In a group of autonomous flying robots, every robot has its own on-board computer and on-board sensors, thus the control of the dynamics is individual-based and decentralized.

Because of these similarities, some of the principles of animal flocking models can be integrated into the control dynamics of autonomous robots [9]. For example, Turgut *et al* presented a dynamical system based on the simplest flocking model and used it to control the motion of so-called Kobots in two dimensions on the ground [10]. In three dimensions, Hauert *et al* presented experiments with fixed-wing agents, as a simple application of two of the three rules postulated by Reynolds (note that there was no true repulsion between the units; they flew at different altitudes) [11].

Thus the principles of flocking models presented above are useful for creating control algorithms for autonomous robots. However, we should not underestimate the ability of animals to maintain highly coherent motion. The prerequisites of smooth collective motion include robustness against reaction times and possible delay in communication, noisy sensory inputs or unpredictable environmental disturbances, such as wind. Animals seem to overcome these difficulties efficiently. However, in robotic systems, these effects can cause unpredictable effects on stability. It is well known, for example that if time delay is present in the communication between swarming agents, instabilities can emerge [12].

One of the main goals of this paper is to provide a model of a general autonomous flying robot integrated into a realistic simulation framework. This model can be used to study the stability of flocking algorithms from the perspective of the deficiencies of realistic systems. The model should contain as many system-specific features as we can take into account, but also should be applicable for many kinds of robots. Due to this 'duality' we define the axioms of the robot model with several independent parameters, corresponding to each source of deficiency in the realistic framework. Specific experimental situations can be realized with a fine-tuned set of these parameters.

Another goal of this paper is to demonstrate that some features of animal flocking models can be useful in collective robotics only if some specific extra aspects of the robots are taken into account. We show that the principles of flocking behaviour can be transformed into unique components of the dynamical system implemented as the control framework of robots. A short-range repulsion is needed to avoid collisions and an implicit viscous friction-like alignment rule term is efficiently used for damping the amplitude of oscillations caused by the imperfections of the system. With simulations and experiments on autonomous quadcopters, we study the stability of two realistic bio-inspired situations: (i) a general self-propelled flocking scenario inside a bounded area and (ii) a collective target tracking setup to reach and smoothly stop at a predefined position.

2. Realistic model of a flying robot

In this section, we present a model of a flying robot based on some features that are general in many realistic robotic systems. In such systems, the motion of the robots is controlled by a low-level algorithm, e.g., a velocity-based PID controller (see appendix A). This low-level control algorithm typically has an input, the desired velocity vector of the actual robot. During flock flights, the time-dependence of the desired velocity of the ith unit can be a function of the positions ($\boldsymbol{x}_i$) and velocities ($\boldsymbol{v}_i$) of the other units:

$$\boldsymbol{v}_i^{\mathrm{d}}(t) = \boldsymbol{f}_i\left(\left\{\boldsymbol{x}_j(t)\right\}_{j=1}^{N}, \left\{\boldsymbol{v}_j(t)\right\}_{j=1}^{N}\right)$$

where N is the number of agents and the $\boldsymbol{f}_i$ function contains the arbitrary features of the controlling dynamics. In an ideal case, the velocity of the ith robot changes to $\boldsymbol{v}_i^{\mathrm{d}}(t)$ at time t immediately. However, a robotic system is never ideal; some of its deficiencies will need to be modelled.

1. *Inertia.* The robots cannot change their attitude or velocity immediately. In general, the desired velocity is an input of a low-level controller algorithm. We assume that in an optimal setup, the system can reach the desired velocity with exponential convergence, with a characteristic time τ_{CTRL}. A simple controller algorithm satisfying this behaviour is the PID controller (see appendix A). The magnitude of acceleration is also limited to $a_{\max}$.
2. *Inner noise.* We have to take into account the inaccuracy of the sensors that provide relative position and velocity information. For example, the uncertainty of the position and velocity measured by a GPS device can be modelled as a stochastic function $\boldsymbol{\eta}_i^{\text{s}}(t)$ (see appendix B). This function can be characterized by a standard deviation σ_{s}. Note that the term 'inner noise' can refer to the inaccuracy of any kind of sensor used in the actual robot system.
3. *Refresh rate of the sensors.* The refresh rate of the sensory inputs fundamentally defines the reaction time and agility of the robots. We consider a limited refresh rate of the sensors: every unit updates sensory data with frequency t_{s}^{-1}. In our current model, t_{s}^{-1} is constant.
4. *Locality of the communication.* The communication between the units has a finite range, r_{c}, thus if the distance between two units is greater than r_{c}, they cannot interact with each other. In other words, the $\boldsymbol{f}_i$ function depends on $\boldsymbol{x}_j$ only if $\left|\boldsymbol{x}_j - \boldsymbol{x}_i\right| < r_{\text{c}}$.
5. *Time delay.* By the time a unit receives and processes position and velocity data from another unit, the data will be old due to data processing and transmission delays. In the simplest approach, time delay can be considered as a constant value, t_{del}.
6. *General noise.* A delta-correlated (Gaussian) *outer noise* term $\boldsymbol{\eta}_i(t)$ with standard deviation σ is added to the acceleration of the units. This term is a model of unpredictable environmental effects such as fluctuations in the wind compensation of the low-level control algorithm.

Considering all the points above, our definition of a realistic system is the equivalent of defining the set $\left\{\tau_{\text{CTRL}}, a_{\max}, r_{\text{c}}, t_{\text{del}}, t_{\text{s}}, \left\{\boldsymbol{\eta}_j(t), \boldsymbol{\eta}_j^{\text{s}}(t)\right\}_{j=1}^{N}\right\}$. Time delay and communication range are hard to measure, can change randomly and have the most dangerous effects on stability. Therefore any kind of $\boldsymbol{f}_i$ has to be investigated with various t_{del} and r_{c} values.

The final form of the model is an equation that defines the acceleration ($\boldsymbol{a}_i(t)$) of each unit:

$$\boldsymbol{a}_i(t) = \boldsymbol{\eta}_i(t) + \frac{\boldsymbol{v}_i^{\text{d}}(t) - \boldsymbol{v}_i(t) - \boldsymbol{v}_i^{\text{s}}(t)}{\left|\boldsymbol{v}_i^{\text{d}}(t) - \boldsymbol{v}_i(t) - \boldsymbol{v}_i^{\text{s}}(t)\right|} \times \min\left\{\frac{\boldsymbol{v}_i^{\text{d}}(t) - \boldsymbol{v}_i(t) - \boldsymbol{v}_i^{\text{s}}(t)}{\tau_{\text{CTRL}}}, a_{\max}\right\}, \qquad (1)$$

$$\boldsymbol{v}_i^{\text{d}}(t) = \boldsymbol{f}_i\begin{pmatrix}\left\{\boldsymbol{x}_j(t - t_{\text{del}}) + \boldsymbol{x}_j^{\text{s}}(t - t_{\text{del}})\right\}_{j\neq i}, \boldsymbol{x}_i(t) + \boldsymbol{x}_i^{\text{s}}(t),\\ \left\{\boldsymbol{v}_j(t - t_{\text{del}}) + \boldsymbol{v}_j^{\text{s}}(t - t_{\text{del}})\right\}_{j\neq i}, \boldsymbol{v}_i(t) + \boldsymbol{v}_i^{\text{s}}(t)\end{pmatrix},$$

where $\boldsymbol{x}_i^{\text{s}}(t)$ and $\boldsymbol{v}_i^{\text{s}}(t)$ give a measure of the integrated position and velocity noise for a random variable $\boldsymbol{\eta}_i^{\text{s}}(t)$, which results from solving the second-order stochastic differential equation $\ddot{\boldsymbol{x}}_i^{\text{s}}(t) = \dot{\boldsymbol{v}}_i^{\text{s}}(t) = \boldsymbol{\eta}_i^{\text{s}}(t)$. In the expression of $\boldsymbol{f}_i$, $\{\ldots\}_{j\neq i}$ denotes a set with iterator $j \neq i$. The function $\boldsymbol{f}_i$ depends on the actual position and velocity of the ith agent and the delayed position and velocity of the other agents and only changes with t_{s}^{-1} frequency. The equations above can be solved using the *Euler* and *Euler–Maruyama* methods.

In the rest of the paper, we choose specific $\boldsymbol{f}_i$ functions with two main features:

(a) $\boldsymbol{f}_i$ depends only on relative coordinates of the interacting units, i.e., no global positioning information is needed in the system:

$$\boldsymbol{f}_i = \boldsymbol{f}_i\begin{pmatrix}\left\{\boldsymbol{x}_j(t - t_{\text{del}}) - \boldsymbol{x}_i(t) + \boldsymbol{x}_j^{\text{s}}(t - t_{\text{del}}) - \boldsymbol{x}_i^{\text{s}}(t)\right\}_{j\neq i},\\ \left\{\boldsymbol{v}_j(t - t_{\text{del}}) + \boldsymbol{v}_j^{\text{s}}(t - t_{\text{del}})\right\}_{j\neq i}, \boldsymbol{v}_i(t) + \boldsymbol{v}_i^{\text{s}}(t)\end{pmatrix}$$

(a) interaction terms in $\boldsymbol{f}_i$ can be expressed as a sum of local pairwise interactions ($\boldsymbol{f}_{ij}$) with other units:

$$\boldsymbol{f}_i = \sum_{j=1}^{N}\boldsymbol{f}_{ij}\left(\tilde{\boldsymbol{x}}_j - \tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{v}}_i, \tilde{\boldsymbol{v}}_j\right)\theta\left(r_{\text{c}} - \left|\tilde{\boldsymbol{x}}_i - \tilde{\boldsymbol{x}}_j\right|\right),$$

where $\theta(x)$ defines the communication range explicitly; it equals 0 if $x < 0$ and equals 1 if $x \geqslant 0$. $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{v}}_i$ are the measured position and velocity values including the modelled inner noise term: $\tilde{\boldsymbol{x}}_i = \boldsymbol{x}_i + \boldsymbol{x}_i^{\text{s}}$ and $\tilde{\boldsymbol{v}}_i = \boldsymbol{v}_i + \boldsymbol{v}_i^{\text{s}}$.

We also choose fixed values for some of the parameters: $t_{\text{s}} = 0.2\text{s}$, $\tau_{\text{CTRL}} = 1$ s, $a_{\max} = 6$ m s^{-2}, $\sigma_{\text{s}} = 0.005$ m^2 s^{-2}. These values represent our state-of-the-art experimental setup with quadcopters. For practical reasons, we saturate the magnitude of desired velocities expressed by the $\boldsymbol{f}_i$ functions at $v_{\max} = 4$ m s^{-1}.

In table 1, we summarize the parameters of the model defined by (1).

3. Self-propelled flocking model

In this section, we present a minimal algorithm that is capable of driving collective robotic systems towards a stable, collision-less, self-organized correlated flocking state. This algorithm is based on the early models of animal swarms [2, 3]. By 'self-organized' we mean that the individuals arrive at a well-defined collective state based on the units' own decisions only [13]. The desired velocity of the agents is now a sum of interaction terms and some extra

Table 1. Parameters of the flying robot model. The column 'Valid range' shows values that are valid for our experimental setup with quadcopters. For further details, see appendices A and B.

Parameter	Unit	Definition	Valid range / value
τ_{CTRL}	s	Relaxation time of low-level controller (e.g., PID controller)	$\tau_{\mathrm{CTRL}} \approx 1$ s
$a_{\max}$	m s^{-2}	Maximum magnitude of acceleration	$a_{\max} = 6$ m s^{-2}
σ_s	m^2 s^{-2}	Measure of inner noise fluctuation	$\sigma_s = 0.005$ m^2 s^{-2}
t_s^{-1}	s^{-1}	Frequency of receiving sensory data	$t_s^{-1} = 5$ s^{-1}
r_c	m	Communication range	$r_c = (30\text{–}140)$ m
t_{del}	s	Time delay of communication	$t_{\mathrm{del}} = (0\text{–}2)$ s
σ	m^2 s^{-3}	Measure of outer noise fluctuation	$\sigma = (0\text{–}0.2)$ m^2 s^{-3}

terms that define the self-propelling behaviour and interactions with a bounded arena. Each term is described below in detail.

We define the agents as self-propelled particles with preferred velocity v_{flock}:

$$\boldsymbol{v}_i^{\mathrm{SPP}} = v_{\mathrm{flock}} \frac{\boldsymbol{v}_i}{|\boldsymbol{v}_i|}. \tag{2}$$

3.1. Short-range repulsion

To avoid collisions, we define a local linear repulsion between the units:

$$\boldsymbol{v}_{ij}^{\mathrm{rep}} = \frac{D\left(\left|\boldsymbol{d}_{ij}\right| - r_0\right)}{\left|\boldsymbol{d}_{ij}\right|} \boldsymbol{d}_{ij} \theta\left(r_0 - \left|\boldsymbol{d}_{ij}\right|\right), \tag{3}$$

where $\boldsymbol{d}_{ij} = \boldsymbol{x}_j - \boldsymbol{x}_i$, D is the strength of the repulsion and r_0 is the interaction range. We consider that the amplitude of fluctuations in the measured position caused by inner noise can be in the same range as r_0. In such a noisy system, the simple linear repulsion is superior to higher-order terms, because errors in the measured position do not cause sudden changes or singularities in the output. If the robots were able to measure their positions more accurately, higher-order terms, like the Lennard-Jones potential, could be used [14].

3.2. Velocity alignment of neighbours

Any kind of velocity alignment rule term in realistic control algorithms should satisfy three assumptions: it should (i) relax the velocity difference of units close to each other, (ii) be local and (iii) have an upper threshold value even when the distances between the units are close-to-zero (similar to the repulsion term). In light of these, we implement the alignment rule with a viscous friction-like interaction term, similarly to [15] and [16]:

$$\boldsymbol{v}_{ij}^{\mathrm{frict}} = C_{\mathrm{frict}} \frac{\boldsymbol{v}_j - \boldsymbol{v}_i}{\left(\max\left\{r_{\min}, \left|\boldsymbol{d}_{ij}\right|\right\}\right)^2} \tag{4}$$

where C_{frict} is the strength of the alignment and $r_{\min}$ defines a threshold to avoid division by close-to-zero distances.

This term is a specific, practical choice for taking into account the tendency of the particles/robots to align their direction of motion. In some sense it is a discrete counterpart of the viscous friction term which would be present in a continuum description such as, e.g., the one considered first by Toner and Tu [17].

The locality of the viscous friction term in practice is guaranteed by the inverse-square decay of the term as a function of distance. However, the maximal velocity $v_{\max}$ and the value of C_{frict} also has to be bounded. The interaction becomes local if the magnitude of $\boldsymbol{v}_{ij}^{\mathrm{frict}}$ becomes negligible compared to $\left|\boldsymbol{v}_i^{\mathrm{SPP}}\right|$ at large distances, i.e., when $C_{\mathrm{frict}} \ll v_{\mathrm{flock}}\left|\boldsymbol{d}_{ij}\right|^2 / 2v_{\max}$ for large values of $\left|\boldsymbol{d}_{ij}\right|$. The optimal ratio of v_{flock} and C_{frict} is thus defined by the limit of the velocities and the desired interaction range.

3.3. Boundaries and shill agents

An important principle of flocking behaviour is a kind of global positioning constraint that contributes to the integrity of the flock. In simulation, this feature of the positioning constraint can be well substituted by using periodic boundary conditions. This is an effective method for examining the large-scale statistical properties of the system. In real experiments, periodic boundary conditions can be imitated by closing the units into a quasi-low-dimensional space, e.g., into a ring-shaped arena [18, 19], but in three dimensions these restrictions are not practical at all.

To study the flocking model with simulations, we placed the units into a square-shaped arena with repulsive walls. We defined the repulsion of the wall as virtual 'shill' agents [20]. If the units are outside the wall, those shill agents try to align the velocities of the units towards the centre of the arena:

$$\boldsymbol{v}_i^{\mathrm{shill}} = C_{\mathrm{shill}} \cdot s\left(\left|\boldsymbol{x}_a - \boldsymbol{x}_i\right|, \tilde{R}\left(\boldsymbol{x}_i, \boldsymbol{x}_a, R\right), d\right) \times \left(v_{\mathrm{flock}} \frac{\boldsymbol{x}_a - \boldsymbol{x}_i}{\left|\boldsymbol{x}_a - \boldsymbol{x}_i\right|} - \boldsymbol{v}_i\right), \tag{5}$$

Table 2. Parameters of the self-propelled flocking algorithm.

Parameter	Unit	Definition
v_{flock}	m s^{-1}	Preferred 'flocking' velocity
D	1 s^{-1}	Strength of repulsion
r_0	m	Interaction range of repulsion
C_{frict}	m^2	Strength of viscous friction
$r_{\min}$	m	This parameter defines a threshold to avoid division by zero
R	m	Side length of the square-shaped arena
C_{shill}		Maximum strength of shill-repulsion near walls
d	m	Characteristic 'width' of the wall

Table 3. Parameters of the target tracking algorithm.

Parameter	Unit	Definition
v_0	m s^{-1}	Preferred velocity far from the target position
r_{CoM}	m	Radius of expected flock size (characteristic size of the meta-agent)
r_{trg}	m	Characteristic size of the target area
d	m	Charascteristic size of the 'transition' area—velocity of the meta-agent approach to zero near the target point with this 'relaxation length'.

where C_{shill} is the strength of the 'shill-repulsion', $\boldsymbol{x}_a$ is the position of the centre of the arena and $s\left(x, R, d\right)$ is a sigmoid curve which smoothly reduces the strength of the repulsion inside the arena:

$$s\left(x, R, d\right) = \begin{Bmatrix} 0 & \text{if} \quad x \in \left[0, R\right] \\ \sin\left(\frac{\pi}{d}\left(x - R\right) - \frac{\pi}{2}\right) + 1 & \text{if} \quad x \in \left[R, R + d\right] \\ 1 & \text{if} \quad x > R + d \end{Bmatrix}. \quad (6)$$

$\tilde{R}$ is a function that defines the shape of the arena (in this case, a square with side length R).

Note that the walls of the arena are pre-defined globally in the simulation, but the repulsive term only depends on the relative coordinates $\boldsymbol{x}_i - \boldsymbol{x}_a$, thus in real robotic systems the arena can be sensed locally, the same way as neighbouring units are.

The sum of the three terms defined above are the minimal prerequisites of flocking behaviour, in other words, with these terms we could guarantee stable and collision-free collective motion in our simulations and experiments:

$$\boldsymbol{v}_i^{\text{d}} = \boldsymbol{v}_i^{\text{SPP}} + \boldsymbol{v}_i^{\text{shill}} + \sum_{j \neq i} \left(\boldsymbol{v}_{ij}^{\text{rep}} + \boldsymbol{v}_{ij}^{\text{frict}}\right) \theta\left(r_{\text{c}} - \left|\boldsymbol{d}_{ij}\right|\right). \quad (7)$$

In table 2, we summarize the parameters of the self-propelled flocking algorithm.

4. Collective target tracking

In this section, we demonstrate that the interaction terms $\boldsymbol{v}_{ij}^{\text{rep}}$ and $\boldsymbol{v}_{ij}^{\text{frict}}$ can be included in other, task-specific control algorithms. We have created a collective target tracking algorithm using an *a priori* defined fixed target point. The algorithm allows the units to perform a smooth transition between two stable states: the flocking state (far from the target) and the collective hovering state (around the target). During this transition near the target point, the preferred magnitude of the velocity has to approach zero smoothly and the coherence and robustness of the flock should be maintained without signs of jamming or oscillations.

Imagine the flock as a 'meta-agent' at the centre of mass moving towards the target position with desired velocity v_0. Each unit must accomplish two tasks without collisions: (i) approach this meta-agent close enough to join the flock and (ii) move parallel with the meta-agent to reach the target collectively.

According to our definition, communication between the robots is local. Therefore, calculating the global centre of mass is physically not possible. Nevertheless, robots can calculate a *local centre of mass* (*CoM*), based on the information available from within their communication range (in a sphere-shaped environment with radius r_{c}). Attraction towards the target point is thus defined as:

$$\boldsymbol{v}_i^{\text{trg}} = v_0 \left[s\left(\left|\boldsymbol{x}_i^{\text{CoM}} - \boldsymbol{x}_i\right|, r_{\text{CoM}}, d\right) \frac{\boldsymbol{x}_i^{\text{CoM}} - \boldsymbol{x}_i}{\left|\boldsymbol{x}_i^{\text{CoM}} - \boldsymbol{x}_i\right|} + s\left(\left|\boldsymbol{x}^{\text{trg}} - \boldsymbol{x}_i^{\text{CoM}}\right|, r_{\text{trg}}, d\right) \frac{\boldsymbol{x}^{\text{trg}} - \boldsymbol{x}_i^{\text{CoM}}}{\left|\boldsymbol{x}^{\text{trg}} - \boldsymbol{x}_i^{\text{CoM}}\right|} \right], \quad (8)$$

where v_0 is the magnitude of the preferred velocity, $\boldsymbol{x}^{\text{trg}}$ is the position of the target, $\boldsymbol{x}_i^{\text{CoM}}$ is the position of the local centre of mass from the viewpoint of the ith agent, r_{trg} is the radius of the target area and r_{CoM} is the radius of the sphere-shaped meta-agent. $s\left(x, R, d\right)$ is the sigmoid function defined in (6). Note that the locality of the viscous friction term defined in (4) depends on the values of v_0 and C_{frict} in this algorithm. Also note that different weights for the target and CoM tracking terms could also be introduced, but we keep these weights at 1 now to keep the algorithm as simple as possible.

The magnitude of the target tracking term saturates at v_0:

$$\tilde{\boldsymbol{v}}_i^{\text{trg}} = \frac{\boldsymbol{v}_i^{\text{trg}}}{\left|\boldsymbol{v}_i^{\text{trg}}\right|} \min\left\{v_0, \left|\boldsymbol{v}_i^{\text{trg}}\right|\right\}. \quad (9)$$

The final desired velocity calculated by the algorithm is:

$$\boldsymbol{v}_i^{\text{d}} = \tilde{\boldsymbol{v}}_i^{\text{trg}} + \sum_{j \neq i} \left(\boldsymbol{v}_{ij}^{\text{rep}} + \boldsymbol{v}_{ij}^{\text{frict}}\right) \theta\left(r_{\text{c}} - \left|\boldsymbol{d}_{ij}\right|\right). \quad (10)$$

In table 3, we summarize the parameters of the target tracking algorithm.

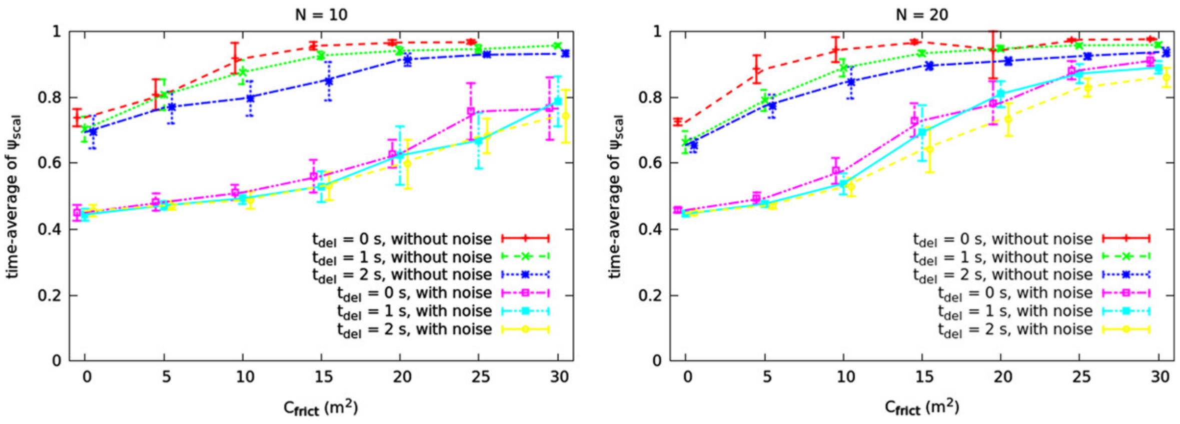

Figure 1. Order parameter of the flocking state $\left(\left\langle \psi_{\mathrm{scal}} \right\rangle_t\right)$ as a function of C_{frict} with 10 and 20 agents. Increasing C_{frict} yields more stably ordered flocks even with relatively large delay and inner and outer noises. Noise parameters are $\sigma = 0.2$ m^2 s^{-3} and $\sigma_s = 0.005$ m^2 s^{-2} in the simulated experiments labelled with 'with noise'. Other parameters are: $D = 1$ s^{-1}, $r_0 = 8$ m, $d = 2$ m, $r_{\mathrm{min}} = 1$ m, $R = 100$ m, $C_{\mathrm{shill}} = 2$, $r_c = 50$ m. Every data point is averaged over 10 simulated experiments with 10 min length and different random initial conditions. Error bars show standard deviation.

5. Results and discussion

In this section, we present realistic simulation and robotic experiment results.

5.1. Simulation of the flocking algorithm

First of all, we demonstrate that typical flocking patterns can emerge even with large delays in the communication and with the presence of inner and outer noise. The coherence of the flocking state can be indicated with the order parameter

$$\psi_{\mathrm{scal}}(t) = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j \neq i} \frac{\boldsymbol{v}_i(t) \cdot \boldsymbol{v}_j(t)}{\left|\boldsymbol{v}_i(t)\right| \left|\boldsymbol{v}_j(t)\right|}, \qquad (11)$$

where N is the number of agents and $\boldsymbol{v}_i(t) \cdot \boldsymbol{v}_j(t)$ is the scalar product of two velocity vectors. In and ideal flocking state, $\psi_{\mathrm{scal}} \approx 1$, while in a disordered state, $\psi_{\mathrm{scal}} \approx 0$. According to figure 1, with lower C_{frict} values, correlated flocking behaviour with high ψ_{scal} cannot be observed. Higher C_{frict} guarantees that the emerged flocking states are stable even in the presence of noise and large time delay.

5.2. Simulation of the target tracking algorithm

The goal of this subsection is to show that the stability of the target tracking algorithm can be guaranteed with our selection of interaction terms used in the flocking algorithm. To study the stability, we analyze two possible quasi-stable states of the system: the flocking state (large velocity, far from the target position) and the hovering state (zero velocity, near the target position). Note that our goal is to show the effects of the interaction terms on the stability, therefore the other parameters (v_0, r_{CoM}, r_{trg} and d) were set to fixed default values. Parameter choice was optimized to guarantee the stable completion of the target tracking task with smooth transition between the flocking and hovering states in an ideal case.

To initialize the flocking state, the units are placed within a 35 m wide square-shaped area 100 m away from the target point. After starting the simulated experiment, in an ideal case, the velocity vectors of the units should become parallel and should have the same magnitude, i.e., stable, ordered flocking behaviour should be observed with $\psi_{\mathrm{scal}} \approx 1$. We define the end of the flocking state when all units are at most $2r_{\mathrm{CoM}}$ far from the target point. After this point, ψ_{scal} shall not be used as an order parameter due to the decreased velocities around the target.

Time delay can reduce the stability of the flocking state (see figure 2), and increasing C_{frict} reduces the strength of instabilities (see the right side of figure 3).

To initialize the hovering state, units are placed around the target point inside a circle with radius r_{trg} and with zero initial velocity. Due to the interaction forces and the attraction towards the target point, in an ideal case, the units will arrange themselves into a lattice-like structure, where the distance between neighbours is approximately r_0. However, if time delay is present in the system, dangerous oscillations can emerge. Since that kind of instability can lead to collisions, it has to be eliminated. The strength of the instability in the hovering state can be described by the average velocity-magnitude:

$$\psi_{\mathrm{vel}}(t) = \frac{1}{N} \sum_{j=1}^{N} \left|\boldsymbol{v}_i(t)\right|. \qquad (12)$$

An increase of $\psi_{\mathrm{vel}}(t)$ represents the growing amplitude and/or frequency of the oscillations.

In the hovering state, the strength of the delay-induced oscillations can be reduced with higher C_{frict} and smaller D values either with or without inner and outer noises (see the left side of figure 3). It is important to note that the instabilities can be reduced by an optimal setup of the interaction parameters only if the sphere around the local centre of mass

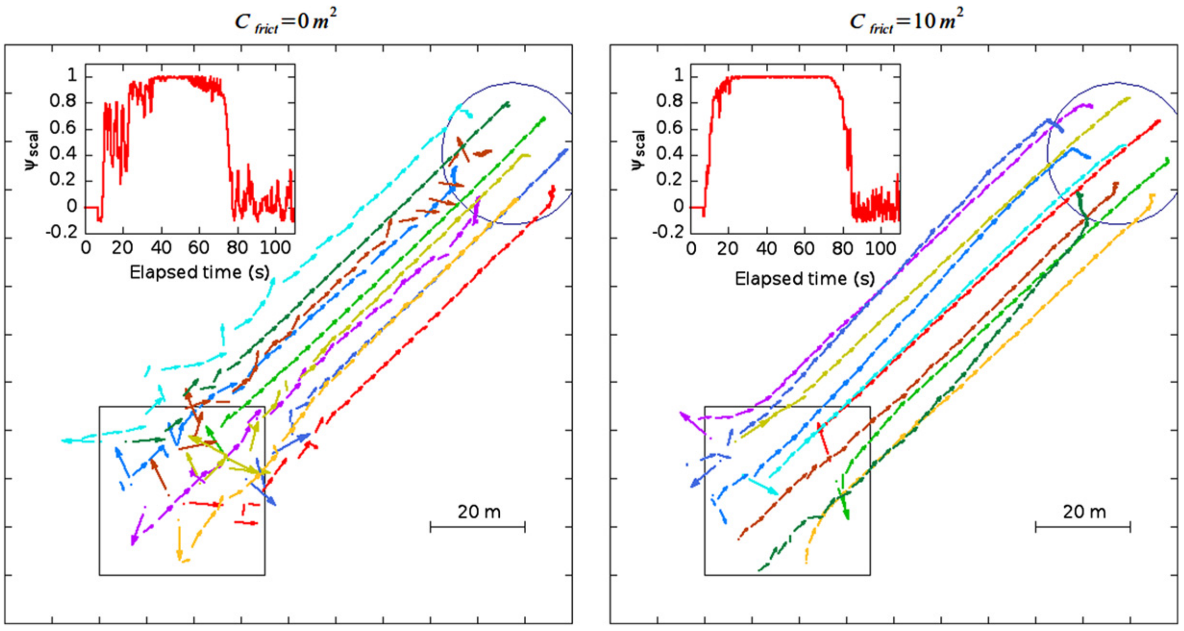

Figure 2. Trajectories of 10 agents with and without viscous friction, without noise, and with $t_{\text{del}} = 1$ s. The arrows represent the actual velocity vectors of units. The insets show the order parameter ψ_{scal} *vs* time. The square represents the starting area; the circle is the environment around the target point with radius $r_{\text{CoM}} = 15.15$ m. Without the friction-like term, oscillations can emerge during the flocking behaviour, which causes severe quasi-stochastic fluctuations in $\psi_{\text{scal}}(t)$. Other parameters are: $D = 1\ \text{s}^{-1}$, $r_0 = 8$ m, $r_{\text{trg}} = 6.5$ m, $d = 2$ m, $v_0 = 2\ \text{m s}^{-1}$, $r_c = 100$ m.

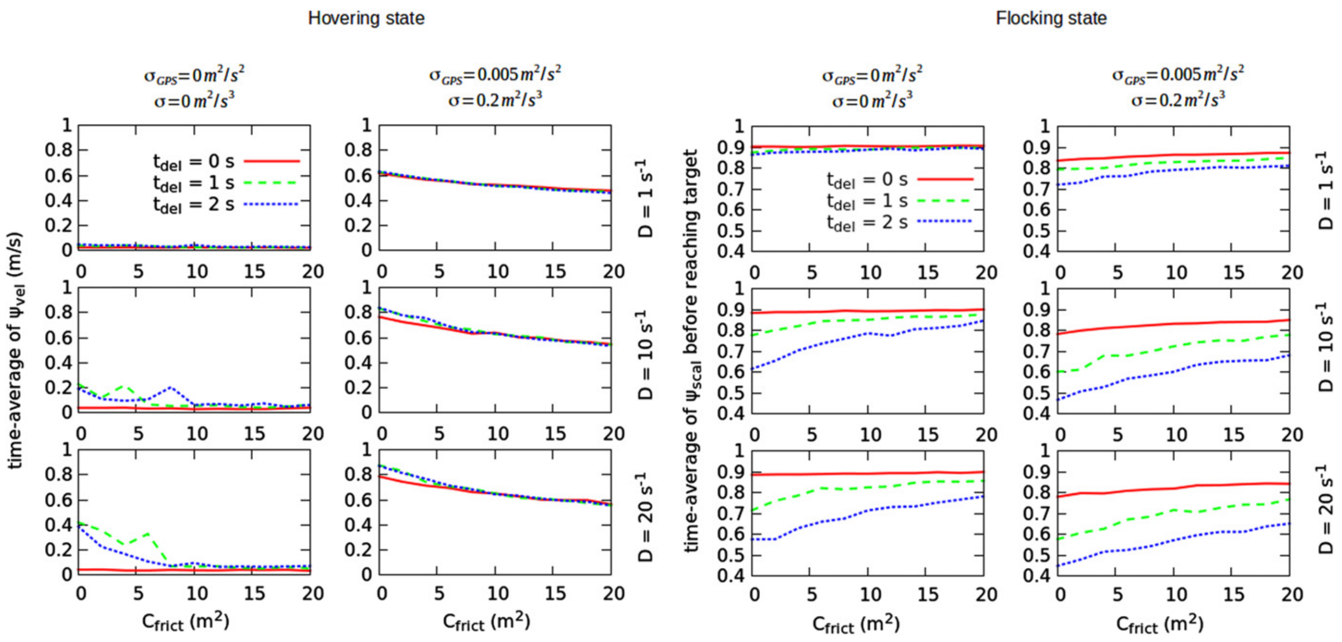

Figure 3. Stability of the two possible states with different D, C_{frict} and t_{del} values. Increasing C_{frict} yields more stable behaviour in all cases. Other parameters are: $N = 10$, $r_c = 100$ m, $v_0 = 2\ \text{m s}^{-1}$, $d = 2$ m, $r_{\text{trg}} = 6.5$ m, $r_0 = 8$ m. r_{CoM} is 12.3 m in noiseless setups, and 13.3 m in setups with non-zero noise level. All data points are averaged over 10 simulated experiments with different random initial conditions.

with radius r_{CoM} is large enough to contain all units with at least the repulsive interaction range (r_0) apart from each other. With giving the units enough space around the target, we can reduce all superfluous excitations caused by the repulsive interactions.

According to figure 3, the additive Gaussian noise term can also reduce the instabilities caused by the time delay in the hovering state. This is a general feature of coupled delayed dynamical systems, since random noise usually acts against synchrony and resonance.

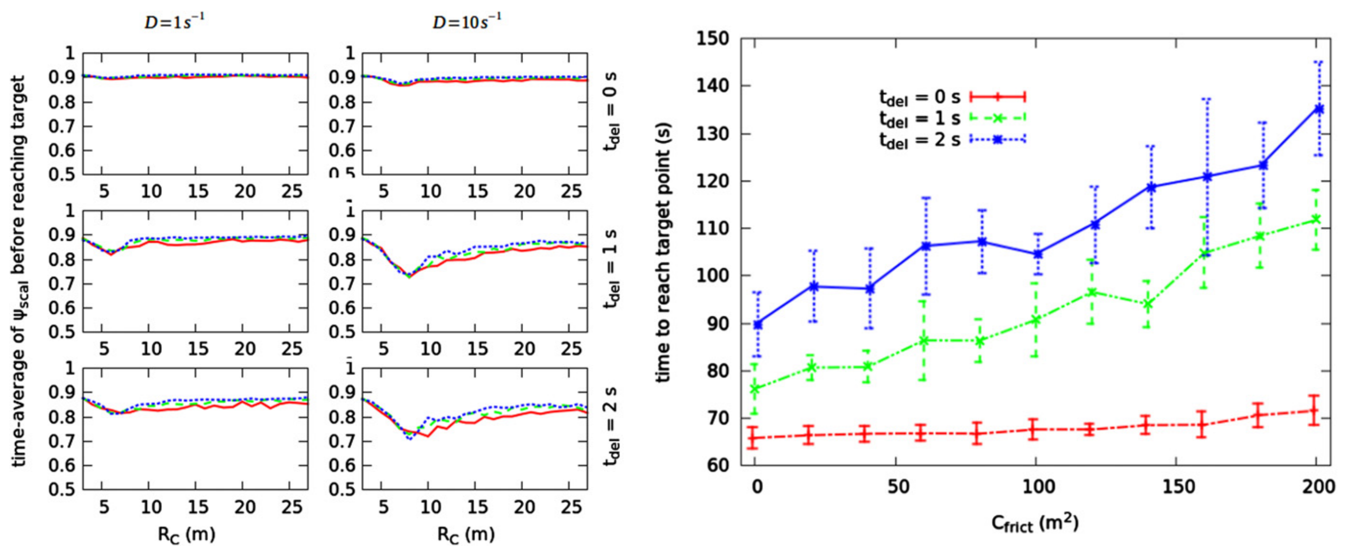

Figure 4. Left: time-average of ψ_{scal} as a function of r_c in the flocking state without noise. Increasing C_{frict} increases the stability of the flocking state even with mid-range (approx. $2r_0$) r_c values. Right: overall time needed to reach target point with various t_{del} and C_{frict} values with r_c = 100 m and D = 1 s^{-1} without noise. Increasing C_{frict} 'slows down' the flock. Other parameters are: N = 10, v_0 = 2 m s^{-1}, d = 2 m, r_{trg} = 6.5 m, r_0 = 8 m. r_{CoM} is approx. 12.3 m. All data points are averaged over 10 simulated experiments with different random initial conditions. Error bars show standard deviation.

One limitation of increasing C_{frict} is that it increases the overall time needed to reach the target point, especially when time delay is present in the system (see the right side of figure 4). With extremely high C_{frict} values, the units with zero initial velocity can practically get stuck at their initial positions.

The effects of the local communication also have to be examined. With small r_c values, the units update their velocity vectors independently, thus each unit aligns its velocity towards the target point. That leads to correlated motion, but collisions can occur due to the lack of communication between the units. When r_c increases close to the range of r_0, units can avoid collisions but they cannot organize themselves into a stable flocking state with high ψ_{scal}. With $r_c \gg r_0$, a correlated collective flocking state can be achieved (see the left side of figure 4).

5.3. Experiments

We have checked the validity of the predictions of our models by implementing the algorithms presented in subsections 3.1 and 3.2 on a flying robotic flock made of 9 quadcopters. Our primary goal was to test the stability of the algorithms under realistic environmental conditions, including, for example, wind of a moderate level and randomly changing direction.

Our robots were based on a quadcopter (Mikrokopter L4, ME) with an on-board Gumstix Overo Water computer. Positions and velocities were measured with U-blox Lea 6-T GPS receivers, and were sent between the robots via XBee Pro modules in broadcast mode (without establishing one-to-one connections or a mesh network). Note that we used GPS for simplicity. GPS is in general not necessary for implementing the described algorithms on real robotic systems. The terms in the control algorithm depend only on the relative positions and absolute velocities. Relative coordinates were calculated using the differences between absolute positions received from GPS devices. Thus, for the time being, we avoided the otherwise difficult issue of the robots sensing the position, heading and velocity of each other with local sensors [21]. The experiments were carried out outdoors, over a large plain field close to Budapest. To analyze the trajectories of the robots, we used data from GPS tracklogs. For further description of our hardware, see [22].

To test the flocking algorithm, we defined a repulsive arena as a square with 120 m sidelength around a global reference point. We performed a 20 m measurement with 9 quadcopters moving freely inside the arena with v_{flock} = 3.5 m s^{-1}. A two minute segment of the successful measurement is presented on the left side of figure 5. Robots performed a correlated motion while crossing the arena and changed to a new direction when they hit the wall. Some minor oscillations emerged near the walls but they always decayed quickly due to the over-damped dynamics introduced with the viscous friction term.

During the test of the target tracking algorithm, the position of the actual target point was broadcasted to the flying robots from a hand-held device in real time. We placed the target point in a car far away from the flock. After take-off, the robots approached the target together and stopped above it with a smooth transition from a tracking to hovering state. After some time of hovering, we drove the car over a straight trajectory and the flock followed it dynamically, still maintaining the stable, grid-like structure. Our results are presented on the right side of figure 5.

We found that our flocking and target tracking algorithm remained stable and safe even in the presence of realistic

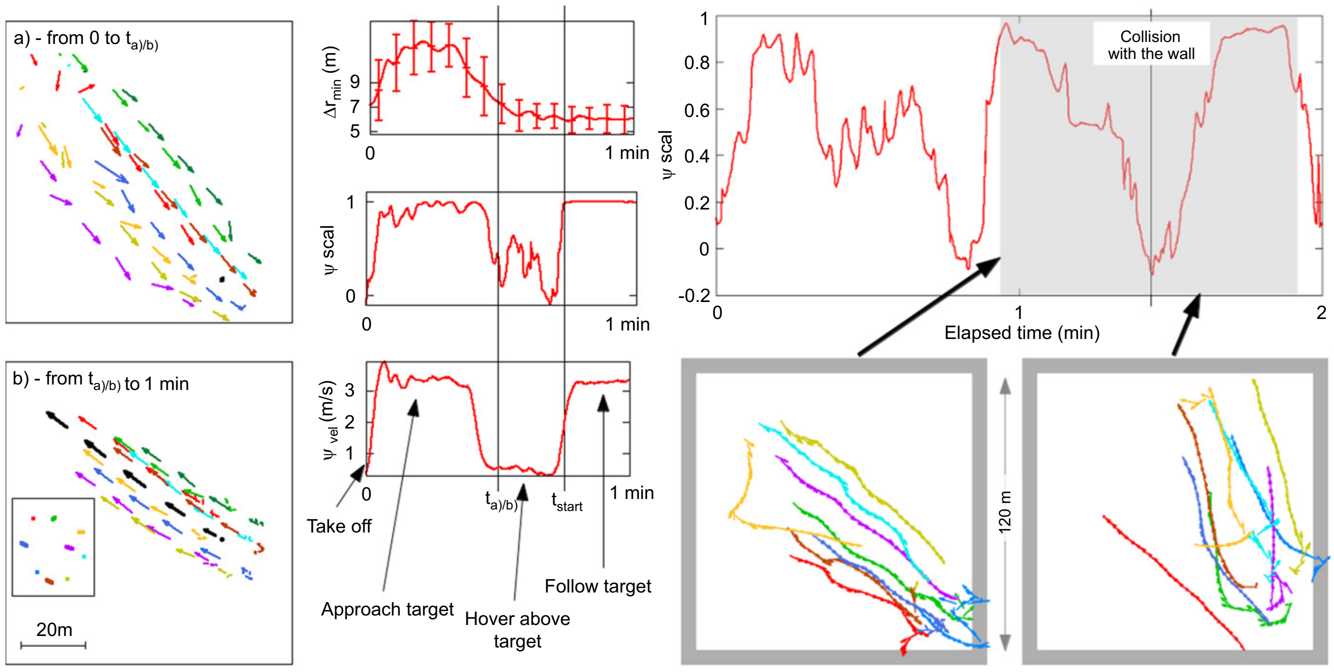

Figure 5. Left: target tracking experiment with autonomous quadcopters. The boxes on the left side show the GPS trajectories of 9 robots and a target point carried by a car (thick black arrows). At the start of the measurement, the target was at a fixed position, and the robots approached it with a smooth transition from flocking state to hovering state (at $t_{a)/b)}$). The inset in box b) shows the grid-like pattern which emerged (with the robots remaining static in the hovering state). At $t_{\rm start}$, the car started to move on a straight trajectory. Next to the trajectories, three of the order parameters are shown. ($\Delta r_{\min}$ is the average distance of the closest neighbours averaged over all the robots). Parameters of the algorithm are: $C_{\rm frict} = 10$ m^2, $D = 1$ s^{-1}, $r_0 = 7$ m, $d = 2$ m, $v_0 = 3.5$ m s^{-1}. Right: experiment of the self-propelled flocking model. At the bottom of the page, the trajectories of eight agents are presented. The grey rectangles represent the arena with repulsive walls. At $t = 1.4$ min, the flock collided with the wall, and after that, it reorganized itself into an ordered state with $\psi_{\rm scal} \approx 1$. Parameters of the algorithm are: $C_{\rm frict} = 20$ m^2, $D = 1$ s^{-1}, $v_{\rm flock} = 3.5$ m s^{-1}, $r_0 = 10$ m.

unpredictable environmental noises. For a demonstration video about some experiments, see the supplementary movie, available at stacks.iop.org/BB/9/025012/mmedia.

5.4. Discussion

In this paper, we presented a realistic simulation framework for developing decentralized control algorithms for swarms of autonomous robots. This framework takes into account several realistic, but not robot-specific features, such as time delay, locality of the communication, inaccuracy and refresh rate of the sensors and inertial effects. Some of these are also present in natural swarming systems, such as bird flocks. We demonstrated the applicability of this framework through the implementation of two algorithms of collective motion: a self-propelled, bio-inspired flocking algorithm and a target tracking setup. Both algorithms contain the same, carefully selected interaction terms inherited from natural flocking models: a repulsive term to avoid collisions and a viscous friction-like velocity alignment rule term for relaxing the velocities of nearby units parallel to each other.

Deficiencies of realistic systems often cause unpredictable instabilities, oscillations and collisions. With simulations, we analyzed the stability of the two algorithms, and found that the instabilities can be reduced with optimal strength of the viscous friction-like term. We implemented the optimized algorithms on a group of real autonomous robots (quadcopters with on-board computers, GPS devices and XBee communication modules). Successful experiments represent a direct proof of the applicability of the model and the algorithms and also show the stability of the algorithms when the system is exposed to unpredictable environmental noises.

Our realistic model can be enhanced in many ways. One major issue is synchronism vs asynchronism in the model and in reality, which appears on many levels: in the update of the simulation iterations, in the modelled delay or in the communication. We have not yet tackled these issues explicitly. However, we always tested the simulation framework with larger delays compared to what was expected in our real system to overestimate the unwanted effects of the delay. Moreover, general random noise terms were introduced in order to compensate for the artificial synchrony of the used delay model. In the future we will certainly enhance our delayed communication model with asynchronous updates. In the real experiments synchrony is not present at any level due to the decentralized control scheme; nevertheless, the simulation results and experimental results are quite similar in general. This indicates that although synchrony is an important element in the simulation framework, its effects are limited in the noisy environment.

In the current setup we used the generally available global positioning system as the most straightforward way of measuring position, velocity and heading. This way, we could concentrate on the development of a functional control framework in a real setup and did not have to deal with any form of 'artificial vision' that is as yet beyond our current knowledge. However, GPS outages could occur at any time due to several independent reasons. In the current model, GPS outages are not modelled explicitly, only through the finite sensor update rates and with the delay in communication. In case of long periods without sensory inputs the system cannot function *per se*. On the other hand, any real application requires robust behaviour. It will be an essential improvement to circumvent this problem when future systems become able to rely on truly local sensory information. In three dimensions this is as yet an unsolved issue; however, we already designed our algorithms to be based only on local data to provide a framework for further, fully autonomous development.

Bio-inspiration was one of the main motivations of our work. Studying the analogies and differences between the behaviour of swarming robotic systems and flocking phenomena in nature reveals many important messages, some of which serve as reverse-bio-inspiration for biological research. For example, we are now inspired to search for additional factors which allow the very highly coherent motion of pigeon flocks, since our experiments suggests that a very short reaction time itself cannot account for the perfectly synchronized flight of many kinds of birds.

Acknowledgement

This work was supported by the FP7 ERC COLLMOT grant. No. 227878., G V was partly supported by EU TÁMOP 4.2.4. A/1-11-1-2012-0001.

Appendix A

PID Controller

To model the specific features of a velocity-based PID controller, we performed measurements with real autonomous flying robots (Mikrokopter L4—ME R/C-controllable quadcopters with a self-developed autopilot board based on a Gumstix Overo Water minicomputer). The low-level controller algorithm implemented on the on-board computer has two inputs: desired (or 'target') velocity and measured velocity. The output of that controller is a control signal value fed to the standard main board of the quadrocopter. The PID loop for controlling velocity is based on the following equation:

$$\varphi_{\mathrm{out}}(t) = K_p e(t) + K_{\mathrm{d}} \frac{\mathrm{d}e(t)}{\mathrm{d}t} + K_i \int_0^t e(t')\mathrm{d}t' + \varphi_{\mathrm{bias}}, \qquad (13)$$

where $e(t)$ is the *error signal*, the difference of the desired and the measured velocity: $e(t) = v^{\mathrm{d}}(t) - v^{\mathrm{m}}(t)$ (v can be the north-south or the east-west component of the velocity vector), the K_p, K_i, K_{d} values are the parameters of the proportional, integral and differential terms and $\varphi_{\mathrm{bias}} = \zeta v^d$ is a feed-forward bias term determined by the linear approximation of the measured velocity as a function of the control signal. We have analyzed the logged data of our robot experiments for finding the proper parameters. The real velocity of the robot is a function of $\varphi_{\mathrm{out}}(t)$ (see the right side of figure A1).

The time-evolution of the real velocity depends on the K_{p}, K_i and K_{d} parameters. In an ideal case, an exponential convergence can be observed with characteristic time τ_{CTRL}. In a non-ideal case, the behaviour is either over-damped with larger settling time or under-damped with oscillations.

Appendix B

GPS device—an example for modelling inner noise

By 'inner noise' we mean the uncertainty of the positions and velocities measured by sensors on the robots. In the case of our quadcopters, the position data is provided by U-blox Lea 6-T GPS receivers. We have made a model to reproduce two main features of the inaccuracy of this device: (i) distribution of the velocity measurement error is close to Gaussian, and (ii) the measured position accuracy is 2.5 m (50% CEP).

We reproduced this fluctuating behaviour of the GPS signal in an empirical model as a particle with Brownian motion in a parabolic potential centred to the real position. The Langevin equation of this situation can be expressed as the second-order stochastic differential equation $\ddot{\boldsymbol{x}}^s(t) = \boldsymbol{\eta}_{\mathrm{s}}(t) = -D_{\mathrm{s}}\boldsymbol{x}^s(t) - \lambda_{\mathrm{s}}\dot{\boldsymbol{x}}^s(t) + \boldsymbol{\xi}(t)$, where $\boldsymbol{\xi}(t)$ is a delta-correlated Gaussian noise term: $\overline{\xi_i(t)\xi_j(t')} = 2\lambda_{\mathrm{s}}\sigma_{\mathrm{s}}\delta(t - t')\delta_{ij}$. To fit the parameters D_{s} and λ_{s}, we have analyzed the fluctuating positional data of a static GPS receiver placed on the ground (for results, see the left side of figure A1). We found that with an optimal setup and the use of the *Euler–Maruyama* method, the simulated GPS position error signal has the same characteristics as the measured data. Note that the fluctuations of position error measured with receivers placed on the ground usually have larger amplitude and frequency than those on a flying quadcopter, thus with this model we over-estimated the real inner noise.

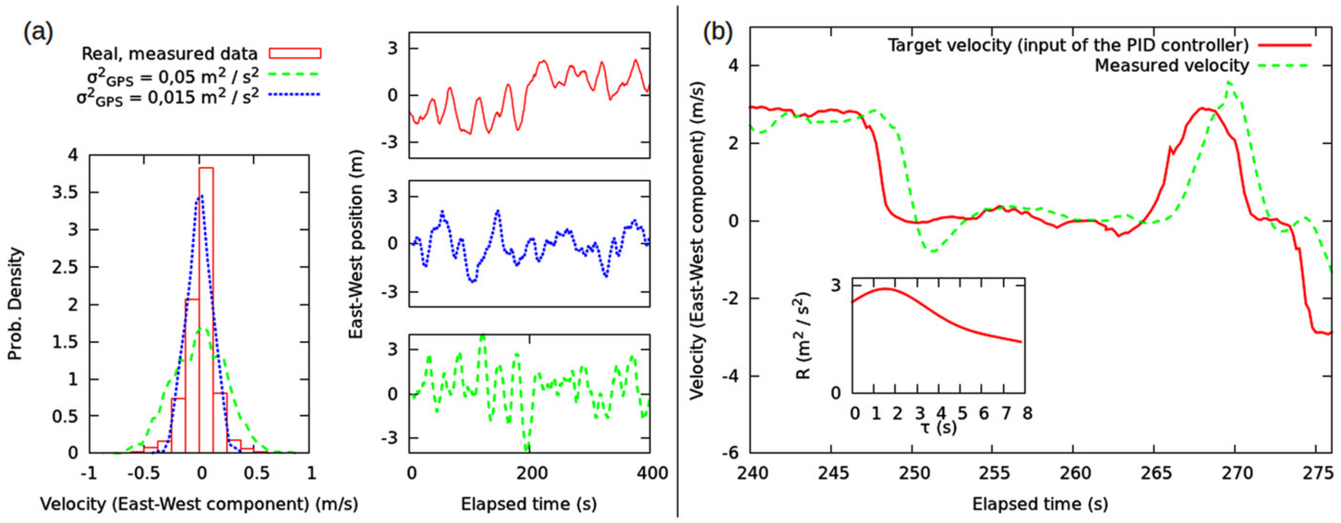

Figure A1. (a) Red (continuous) line: East–West position versus time (right) and velocity distribution (left) measured with a GPS device. The measurements were 20 min long. Green and blue (dotted/dashed) lines: positions as functions of time and velocity distribution modelled with different parameter settings. The parameters of the 'blue/dotted' case are: $\sigma_s = 0.005\ \mathrm{m}^2\,\mathrm{s}^{-2}$, $\lambda_s = 0.1\ \mathrm{s}^{-1}$, $D_s = \sqrt{0.2\sigma_s}\,(3\ \mathrm{m})^{-1}$.
(b) Comparison of the desired ('target') velocity and the real, measured velocity of a real quadrocopter (an extract from a longer measurement is shown). The inset shows the cross-correlation function of the two curves. The relaxation time of the PID controller is at the maximum of this function. In that case, $\tau_{\mathrm{CTRL}} \approx (1\text{–}2)\mathrm{s}$.

References

[1] Vicsek T and Zafeiris A 2012 Collective motion *Phys. Rep.* **517** 71–140
[2] Reynolds C W 1987 Flocks, herds and schools: a distributed behavioural model *SIGGRAPH Comput. Graph* **21** 25–34
[3] Vicsek T, Czirók A, Ben-Jacob E, Cohen I and Shochet O 1995 Novel type of phase transition in a system of self-driven particles *Phys. Rev. Lett.* **75** 1226–9
[4] Szabó P, Nagy M and Vicsek T 2009 Transitions in a self-propelled-particles model with coupling of accelerations *Phys. Rev.* E **79** 021908
[5] Dong H, Zhao Y and Gao S 2013 A fuzzy-rule-based couzin model *J. Control Theor. Appl.* **11** 311–5
[6] Couzin I D, Krause J, Franks N R and Levin S A 2005 Effective leadership and decision-making in animal groups on the move *Nature* **433** 513–6
[7] Grossman D, Aranson I S and Jacob E B 2008 Emergence of agent swarm migration and vortex formation through inelastic collisions *New J. Phys.* **10** 023036
[8] Szabó B, Szőllősi G J, Gönci B, Jurányi Zs, Selmeczi D and Vicsek T 2006 Phase transition in the collective migration of tissue cells: Experiment and model *Phys. Rev.* E **74** 061908
[9] Brambilla M, Ferrante E, Birattari M and Dorigo M 2013 Swarm robotics: a review from the swarm engineering perspective *Swarm Intell.* **7** 1–41
[10] Turgut A, Çelikkanat H, Gökçe F and Şahin E 2008 Self-organized flocking in mobile robot swarms *Swarm Intell.* **2** 97–120
[11] Hauert S, Leven S, Varga M, Ruini F, Cangelosi A, Zufferey J C and Floreano D 2011 Reynolds flocking in reality with fixed-wing robots: communication range vs. maximum turning rate *IROS 2011: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems* pp 5015–20
[12] Forgoston E and Schwartz I B 2008 Delay-induced instabilities in self-propelling swarms *Phys. Rev.* E **77** 035203
[13] Floreano D and Mattiussi C 2008 *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies* (Cambridge, MA: MIT Press) section 7.1 p 516
[14] Ferrante E, Turgut A E, Huepe C, Stranieri A, Pinciroli C and Dorigo M 2012 Self-organized flocking with a mobile robot swarm: a novel motion control method *Adapt. Behav.* **20** 460–77
[15] Cucker F and Smale S 2007 Emergent behavior in flocks *IEEE Trans. Autom. Control* 852–62
[16] Helbing D, Farkas I and Vicsek T 2000 Simulating dynamical features of escape panic *Nature* **407** 487–90
[17] Toner J and Tu Y 1995 Long-range order in a two-dimensional dynamical *XY* model: how birds fly together *Phys. Rev. Lett.* **75** 4326–9
[18] Yates C A, Erban R, Escudero C, Couzin I D, Buhl J, Kevrekidis I G, Maini P K and Sumpter D J T 2009 Inherent noise can facilitate coherence in collective swarm motion *Proc. Natl Acad. Sci. USA* **106** 5464–9
[19] Tarcai N, Virágh C, Ábel D, Nagy M, Várkonyi P L, Vásárhelyi G and Vicsek T 2011 Patterns, transitions and the role of leaders in the collective dynamics of a simple robotic flock *J. Stat. Mech.* **2011** P04010
[20] Han J, Li M and Guo L 2006 Soft control on collective behavior of a group of autonomous agents by a shill agent *J. Syst. Sci. Complex.* **19** 54–62
[21] Moeslinger C, Schmickl T and Crailsheim K 2011 A minimalist flocking algorithm for swarm robots *Advances in Artificial Life: Darwin Meets von Neumann* (Berlin: Springer)
[22] Vásárhelyi G, Virágh C, Tarcai N, Szörényi T, Somorjai G, Nepusz T and Vicsek T 2014 Outdoor Flocking and Formation Flight With Autonomous Aerial Robots *IROS 2014: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems* submitted