Disclaimer: This work has been submitted for publication to the IEEE IROS 2014 Conference. Copyright with IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE. This material is presented to ensure timely dissemination of scholarly and technical work. Copyright and all rights therein are retained by authors or by other copyright holders. All persons copying this information are expected to adhere to the terms and constraints invoked by each author’s copyright. In most cases, these works may not be reposted without the explicit permission of the copyright holder. For more details, see the IEEE Copyright Policy.

Outdoor flocking and formation flight with autonomous aerial robots*

G. Vásárhelyi, Cs. Virágh, G. Somorjai, N. Tarcai, T. Szörényi, T. Nepusz, T. Vicsek

***Abstract*— we present the first decentralized multi-copter flock that is capable of stable autonomous outdoor flight. By autonomous we mean that all members navigate themselves based on the dynamic information received from other robots in the vicinity, without any central data processing or control. Instead, all the necessary computations are carried out by miniature on-board computers. The only global information the system exploits is from GPS receivers, while the units use wireless modules to share this positional information with other flock members locally. Collective behavior is based on a decentralized control framework with bio-inspiration from statistical physical modelling of animal swarms. The model allows for stable group flight even in noisy, windy, delayed and error-prone environment. Using this framework we successfully demonstrated several fundamental collective flight tasks with up to 11 units: i) we achieved self-propelled flocking in a bounded area with self-organized object avoidance capabilities and ii) performed collective target tracking with stable formation flights (grid, rotating ring, straight line). With realistic numerical simulations we demonstrated that the local broadcast-type communication and the decentralized autonomous control method allows for the scalability of the model for larger flock sizes.**

INTRODUCTION

Collective motion is one of the most spectacular phenomena in nature, where the local behavior of many autonomous, similar individuals results in complex motion patterns, creating a fluctuating meta-organism [1]. Besides being truly astonishing, flocking has many advantages in natural systems, since a group of animals or people is: i) more effective when solving foraging or navigational tasks [2]; ii) robust in terms of the knowledge the flock holds about its actual goal due to redundancy and the system's intrinsic parallel-processing structure; iii) more alert regarding environmental threats and more defensive against attacks [3] [4]. Such natural systems provide bio-inspiration for swarm robotics aiming at creating nature-like, i.e., efficient flocks of artificial units with decentralized control.

There are already a great number of two-dimensional ground-based robotic swarms, being able to perform intriguingly complex tasks, such as playing soccer [5], carrying objects together [6], being self-organized [7], performing self-assembly [8] or simply being large-scale [9] [10]. Implementing robots capable of moving in three-dimensions has a great extra potential. However, achievements concerning aerial flocks have been rather limited so far. This fact comes partly from the current technological difficulties of creating even one reliable flying object that is capable of autonomously performing all the necessary flight maneuvers, such as stable hovering, flying at predetermined speed in a given direction or adapting to a time-dependent altitude and/or flight direction quickly enough. However, the recent appearance of small-scale, lightweight flying robots, commonly named as multi-copters, seems to be filling this niche perfectly. Another severe technical difficulty is the proper perception of the environment and other flock members, i.e., sensing absolute or relative positions reliably. All current solutions to this challenge are limited and depend on state-of-the-art technology: computer-vision, infrared, ultrasonic, laser or digital wireless. In order to overcome some of the constraints of these approaches, we have chosen a combination of GPS devices and wireless communication to obtain local relative positional data. In this way we could focus on our primary goal, the implementation of our control algorithms into a real outdoor system.

There have been several remarkable attempts for the creation of 3D flying flocks. Welsby et al. used three motorized balloon-like objects that were wandering in the air together indoor [11]. De Nardi and Holland proposed (but did not create) a 3D flock of helicopters [12]. Hauert et al. 2011 presented an autonomous flock of 10 fixed-wing UAVs flying at relatively large distance from each other [13]. Their system is a noteworthy example of an autonomous Reynolds-flock [15], although their UAVs did not have the eventual constraint of avoiding each other since they flew at different fixed altitudes. As we will see, repulsion between units is the origin of self-excited oscillations that are quite challenging to handle. Stirling et al. stated that they created an autonomous indoor quadrotor flock of 3 units. In fact, two of the robots were attached to a ferromagnetic ceiling and only one unit was flying at a time, using the other two as reference points for navigation [16]. Kushleyev et al. created the most remarkable flock of 20 miniature quadcopters, but their system is not autonomous by our definition since a central computer calculates the navigational instructions for all robots using an external, fixed indoor positioning system [17]. Bürkle et al. created an outdoor quadcopter swarm, but with intensive central processing at a ground station [18].

*Research supported by the EU ERC COLLMOT project.

G. V. Author was partly supported by EU TÁMOP 4.2.4.A/1-11-1-2012-0001.

All authors are with the Department of Biological Physics, Eötvös University, Budapest, Hungary (corresponding e-mail: vasarhelyi@angel.elte.hu)

G. V., G. S., T. N. and T. V. Authors are with the MTA-ELTE Research Group for Statistical and Biological Physics, Budapest, Hungary

In this paper we present the first outdoor 'GPS-vision' based swarm of autonomous flying robots with decentralized self-control and stable self-organization capabilities. Our approach was to use commercially available, affordable, open source products as base units, already having a low-level control of flight stability. We have expanded these units with our self-developed autopilot board and custom-designed flocking algorithms for the high-level control of the units within the swarm.

Structure of a Unit

A. Basic Unit

Our flying robots are based on a partially open source quadcopter, called MK Basicset L4-ME, from MikroKopter Co., Germany [19] (Fig. 1), capable of performing onboard self-stabilization and altitude hold. In its original form, it is controlled manually with a standard R/C remote controller.

The main board of the copter accepts virtual controller signals (pitch/roll/yaw and altitude setpoint) from an arbitrary extension board; we used this channel to gain full automatic control over the unit, while maintaining the option of manual interruption with the remote controller.

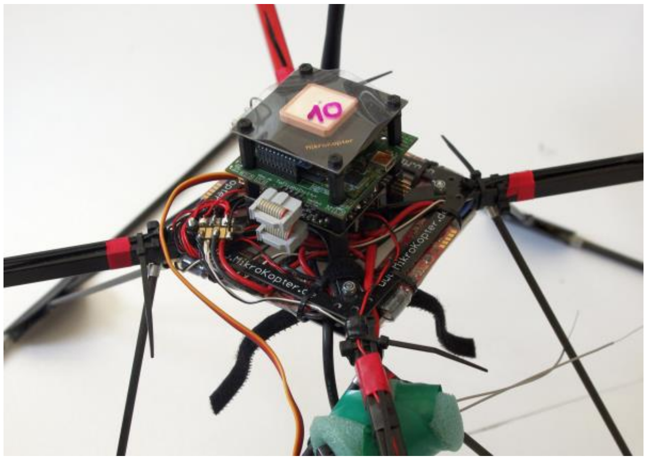

Figure 1. Image of a unit with the commercial base copter below, the custom made (green) extension board above and the GPS on top. Take-off mass is around 1 kg, tip-to-tip diameter is 80 cm. We achieved a maximum of 20 minutes flying time by using 3000mAh, four-cell LiPo batteries.

B. Processor Board

We have developed an extension board (called FlockControl) that contains a 3D gyroscope, a 3D accelerometer, a 3D magnetometer, a pressure sensor, a GPS receiver, a 2.4GHz XBee unit (for digital wireless communication) and a GumStix Overo Water mini-computer with standard Linux OS. The extension board connects to the main board through SPI. It receives information from the base unit about error conditions and inner state variables (attitude, heading, battery level, remote controller signals) and communicates with other flock members via XBee (sharing ID, position, velocity, attitude and status info). The extension board feeds these inputs into the flocking algorithm and sends the calculated virtual steering signals to the low level control board. The main refresh rate of the steering signals is 40 Hz (Fig. 2), individual GPS refresh rate is 5 Hz and XBee communication rate is 10 Hz.

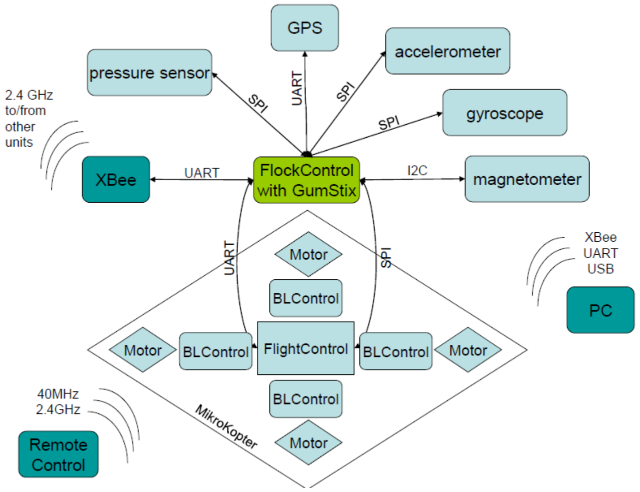

Figure 2. The main structure of one unit. Communication methods between parts are indicated on the arrows.

C. Communication

The XBee modules work in broadcast sending mode, units send data without establishing one-to-one connection or mesh network with other units. Flock members process incoming XBee packets only from other robots, which are inside their communication range (typically around 50–100 m). That is similar to the way e.g. birds fly in a flock, mostly using the information from neighboring birds [20]. This locality of the communication is a simple way to keep the network scalable to larger flock sizes.

D. Velocity Control

Automatic steering signals are calculated in a world reference frame. The three axes of the control (North, East, Down) are independent from each other, due to the structure of the quadcopter. In the vertical direction we rely on the original altitude-hold algorithm of the L4-ME copter that maintains a pre-defined altitude during most of the flight. Autonomous self-organization thus takes place in a quasi-two-dimensional space, which is satisfactory for many applications, like area coverage or distributed search. During the development phases we defined a slightly different altitude set point for all copters (in the 5–15 m range above ground) to have a safety height distance between units, in case unexpected horizontal control errors would occur. However, we managed to carry out the latest flights with all units at the same altitude, with true repulsion between units in the horizontal plane.

For the two horizontal axes we implemented a standard PID based velocity control algorithm that converts the target velocity output of the flocking algorithm into the virtual steering signals. Two independent PID controllers are used for the two global axes (North and East) and the target steering signal is converted into a body-fixed coordinate system based on the heading information calculated from the signals of the magnetometer and the other inertial sensors.

Since the *P* and *D* terms of the velocity PID tend to zero as the measured velocity is getting closer to the target velocity, the *I* term must maintain the necessary output signal level for constant speed flight. When flying in a tight flock, quick response is essential; therefore, we have added a linear feed-forward (also known as bias) term to the control loop, by which the system can predict the magnitude of the necessary adjustment based on earlier experience. This way, the *I* term only needs to fine-tune the estimated stick response and thus the relaxation time of the control loop is significantly reduced. To minimize oscillations of an individual copter, we tuned the PID for slower responsiveness and avoided overshoot of the control.

The performance of the velocity controller was tested in a mobile target tracking setup and it was found that the system is stable and performs well even in moderate wind (up to 5 m/s) with an approximately 1.5 s overall time lag. It is important to note that the PID loop is responsible only for a minor portion of this lag. Other delay-inducing factors are the reaction time of the base unit to the steering signal, the low GPS update rate and the limited acceleration due to the inertia of the copters (Fig. 3).

A price for the stable and overshoot-free, but slower control is paid back at the flock's level, where interesting chaotic oscillations can emerge due to delays, similarly to ghost traffic jams on highways, caused by the slow reaction time (~1s) of humans [21]. These oscillations had to be handled by our flocking algorithm.

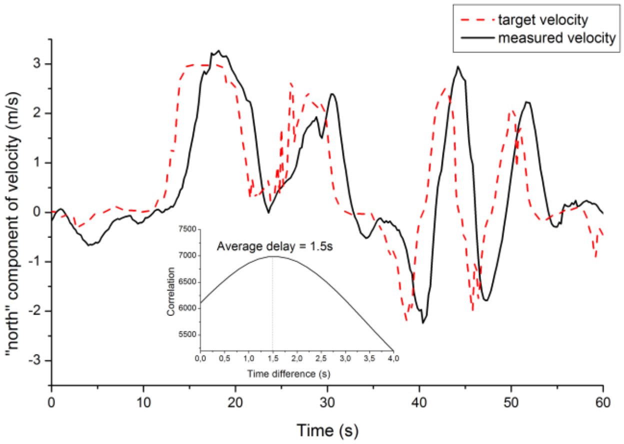

Figure 3. Snapshot of a general target tracking measurement with PID velocity control (*P*=30, *D*=30, *I*=2). Main figure shows the measurement in time, the inset shows the correlation between target velocity (the input of the PID controller) and measured velocity (the final output of the system) as a function of time difference between them. An average time lag of around 1.5 s is present in the control system even if tuned well, due to PID relaxation time, base unit reaction time and inertia limited acceleration.

Other System Components

Ground Control

We have also developed a ground station console application for real-time monitoring and debugging purposes. The application runs on an external computer and gathers general telemetry information from all flock members through XBee. Besides monitoring the state of the flock, the ground station can be used for several other purposes, like requesting debugging packets from individual copters with detailed information, changing system parameters in flight to tune behavior, batch uploading new settings, simulate errors or force landing of individuals or all units at once, too. These are all essential features when one works with a large group of robots at once.

Simulation Framework

We have developed a realistic simulation framework to test various flocking algorithms and to optimize their parameter space before flying with real robots. One main goal of the simulations was to investigate the effects of all sorts of delays present in the system, i.e., reaction time arising from communication and data processing and relaxation time of the controller. A flocking algorithm had to be found that is not sensitive to large delays in terms of stability, in other words, in which delays do not generate undesired oscillations. We combined several techniques to achieve this goal, such as using smooth functions instead of sharp ones, using slack in potential valleys or using special over-damped dynamics in the flocking algorithms.

The simulation framework also takes into account many of the inaccuracies of a real system: the position and velocity measurement error, general Gaussian noise corresponding to the environmental effects, the low update rate of the GPS and the limited range of the communication. The final algorithms had to remain stable in that error-prone, realistic framework, as well. More details about the simulation framework and the stability optimization of the algorithms are discussed in a separate paper [14].

Decentralized Control Algorithm

The control algorithm has a structure analogous to that of the first flocking model of Reynolds [15]: a self-propelled particle needs short-range repulsion from other units, medium-range velocity alignment with neighbors and a global positional constraint to remain with the flock. However, finding the best candidate terms for these tasks in a real environment is not trivial and could be application dependent.

Below we specify how the algorithm described in [14] was implemented for the real flocking and formation flights in the present work. The basic model we use consists of the following main parts (upper indices in the equations mark individuals in the flock).

Short-range Repulsion: Pair Potentials

A repulsive distance-based potential acts between all close units to avoid collisions:

$$\vec{a}^{i}_{pot} = \begin{cases} -D \sum_{j \neq i} min\left(r_1,\ r_0 - \left|\vec{x}^{ij}\right|\right) \frac{\vec{x}^{ij}}{\left|\vec{x}^{ij}\right|}, \\ \qquad \text{if } \left|\vec{x}^{ij}\right| < r_{0,} \\ \qquad 0 \text{ otherwise,} \end{cases} \tag{1}$$

where D is the spring constant of a repulsive half-spring, $\vec{x}^{ij} = \vec{x}^{j} - \vec{x}^{i}$ is the difference of positions of unit *i and j*, r_0 is the equilibrium distance, i.e., the distance above which there is no repulsion between units and r_1 is used to define an upper threshold for the repulsion to avoid over-excitation of units.

In time-lagged systems with even small reaction time, potential functions in the control can be the source of self-excited oscillations [22]. Even though we used only linear and limited repulsive terms, oscillations emerged in some situations. To avoid these, we forced over-damped dynamics through the next term, the velocity alignment.

Middle-range Velocity Alignment: Viscous Friction

Units close to each other damp their velocity *difference* to reduce oscillations and to synchronize collective motion with a viscous friction-like term, similar to the one in [23] or [24]:

$$\vec{a}^{i}_{slip} = C_{frict} \sum_{j \neq i} \frac{\vec{v}^{ij}}{\left(max\left(\left|\vec{x}^{ij}\right| - (r_0 - r_2),\ r_1\right)\right)^2}, \qquad (2)$$

where C_{frict} is the viscous friction coefficient, $\vec{v}^{ij} = \vec{v}^{j} - \vec{v}^{i}$ is the velocity difference between units *i* and *j*, r_2 defines a constant slope around the equilibrium distance r_0, finally, r_1 defines a threshold again to avoid division by close-to-zero distances due to e.g. measured GPS position error.

Note that equations 1 and 2 are both dependent of r_0 to allow for the dynamic tuning of flock density. This is essential when e.g. one needs to extend the model to higher velocity ranges, where larger 'breaking distance' is needed between units.

Global Positional Constraint I: Flocking

One way of keeping individual robots together is to define a bounded area around a global reference point and implement a general self-propelled flocking model [20] that makes the units move around within the walls of the area. The self-propelling term is defined as

$$\vec{v}^{i}_{spp} = v_{flock} \frac{\vec{v}^{i}}{\left|\vec{v}^{i}\right|}, \qquad (3)$$

where v_{flock} is a constant flocking speed the units try to maintain. The bounding walls are pre-defined globally, but they appear as local attractive shill agents [25] that try to pull units back towards the center of the flight area through virtual velocity alignment:

$$\vec{a}^{i}_{wall} = C_{shill} f\left(\left|\vec{x}_{trg} - \vec{x}^{i}\right|, R, d\right)\left(v_{flock} \frac{\vec{x}_{trg} - \vec{x}^{i}}{\left|\vec{x}_{trg} - \vec{x}^{i}\right|} - \vec{v}^{i}\right), \qquad (4)$$

where C_{shill} is the viscous friction coefficient of the wall, $\vec{x}_{trg}$ is the center of the flight area, i.e., the position of a global reference/target point and *f(x)* is a smooth transfer function:

$$f(x, R, d) = \begin{cases} 0 & if\ x \in [0, R] \\ \frac{\sin\left(\frac{\pi}{d}(x - R) - \frac{\pi}{2}\right) + 1}{2} & \text{if } \mathrm{x} \in [R, R + d] \\ 1 & if\ x \in [R + d, \infty] \end{cases} \qquad (5)$$

where *R* is set to be the distance between the reference point and the wall and *d* defines the width of the decay, i.e., the softness of the wall.

With this definition the soft-wall interacts with the units in a smooth way, only when they are outside of the flight area. Object avoidance in general can also be introduced in a similar way with *repulsive* shill agents.

Note that in the above model all interaction terms are local or quickly decay with distance, corresponding to the short range of the communication. Scalability to larger flock sizes is analyzed in realistic numerical simulations.

Global Positional Constraint II: Formation Flights

Another option for global attraction can be introduced without explicit self-propelling, through formation flights around a (static or dynamic) global reference/target point. For this, we split target tracking into two parts:

$$\vec{v}^{i}_{track} = \begin{cases} v_0 \frac{\vec{v}^{i}_{shp} + \vec{v}^{i}_{trg}}{\left|\vec{v}^{i}_{shp} + \vec{v}^{i}_{trg}\right|} \ if \left|\vec{v}^{i}_{shp} + \vec{v}^{i}_{trg}\right| > v_0 \\ \vec{v}^{i}_{shp} + \vec{v}^{i}_{trg}\ otherwise \end{cases}, \qquad (6)$$

where $\vec{v}^{i}_{shp}$ defines how units arrange themselves into shapes relative to the individually calculated center of mass:

$$\vec{v}^{i}_{shp} = \beta v_0 f\left(\left|\vec{x}^{i}_{shp} - \vec{x}^{i}\right|, R, d\right) \frac{\vec{x}^{i}_{shp} - \vec{x}^{i}}{\left|\vec{x}^{i}_{shp} - \vec{x}^{i}\right|}, \qquad (7)$$

while $\vec{v}^{i}_{trg}$ defines how the center of mass follows the target with smoothly adjusted, variable speed:

$$\vec{v}^{i}_{trg} = \alpha v_0 f\left(\left|\vec{x}_{trg} - \vec{x}^{i}_{COM}\right|, R, d\right) \frac{\vec{x}_{trg} - \vec{x}^{i}_{COM}}{\left|\vec{x}_{trg} - \vec{x}^{i}_{COM}\right|}. \qquad (8)$$

Constants α and β in the range of [0,1] define the strength of the velocity components relative to v_0, a maximal tracking velocity, $\vec{x}^{i}_{COM}$ is the locally calculated center of mass, $\vec{x}^{i}_{shp}$ is the desired position in the formation around $\vec{x}^{i}_{COM}$ and *f(x,R,d)* is the smooth transfer function introduced in equation 5 to avoid abrupt changes in the dynamics.

For basic shapes, like a straight line, a (rotating) circle or a simple lattice-like grid, one piece of global information is needed: N, the number of robots, which defines the size of the formation. The equations defining $\vec{v}^{i}_{shp}$ are the following:

i) for a simple *grid*, we have $\vec{x}^{i}_{shp} = \vec{x}^{i}_{COM}$ and $R = \frac{r_0}{2} g(N) - \frac{r_0}{2}$, where *g(N)* is a heuristic function defining the radius of the smallest circle that can contain *N* unit circles [26]. This way all units can fit in the target area around the center of mass of the flock tightly packed. The grid-like arrangement is settled due to the pairwise repulsion and the velocity alignment.

ii) for a *ring*, we define $\vec{x}^{i}_{shp}$ as the intersection of the ring around $\vec{x}^{i}_{COM}$ and the bisectrix of the two closest neighbors defined with a simple angle-based signed metric around the center of mass. The radius of the ring has to be defined to fit N robots on its contour. *R* is set to a small value or zero as the transfer function is used for individual position tracking now. A rotating ring can be achieved with a simple self-propelling term in the tangential direction relative to $\vec{x}_{COM}$. The direction of rotation can be chosen in a self-organized manner according to the direction of the average tangential velocity of all units.

iii) for a *line* we define the signed neighborhood metric as the projected distance on the line from $\vec{x}^{i}_{COM}$ and set $\vec{x}^{i}_{shp}$ as the average position of the two neighbors on the line. Units with only one or zero neighbors must approach the closest end of the line to ensure that units settle with the equilibrium distance (r_0) between them. R is set to a small value or zero again. The angle of the line can be defined in a self-organized way from a linear fit of actual positions.

Full Dynamic Equation

The final form of the velocity evolution is similar to the so-called optimal velocity model describing traffic jams [21] or escape-panic [27]:

$$\vec{v}^i(t+\Delta t) = \vec{v}^i(t) + \frac{1}{\tau}\left(\vec{v}^i_{spp} + \vec{v}^i_{track} - \vec{v}^i(t)\right)\Delta t + \left(\vec{a}^i_{pot} + \vec{a}^i_{slip} + \vec{a}^i_{wall}\right)\Delta t, \tag{9}$$

where τ is the characteristic time needed to reach the velocity $\vec{v}^i_{spp} + \vec{v}^i_{track}$.

Equation 9 is used as the main differential equation in the numeric simulation and also as the main function determining the desired velocity of a copter. Note that in equation 9 the SPP flocking and formation-flight models are mixed, but they are usually used separately, depending on the actual task of the flock. Separation is achieved by setting v_{flock} and v_0 to the appropriate value.

To compensate for the time-lag previously described, the PID controller input $\vec{v}^i(t+\Delta t)$ is always calculated for a time instance far ahead t. In other words we tell the system in the present what to prepare for in the future, allowing it to react in time. In the simplest linear approximation this is achieved by selecting a large Δt (in the magnitude of 1 s or so).

Note that the final target velocity is always restricted in the simulations as well as in the real flocking algorithm. This limits the speed of the units for safety reasons.

Table I summarizes optimal parameter choice for the algorithms.

Results

We successfully established the first decentralized, autonomous multi-copter flock in an outdoor environment, with swarms of up to 10 flying robots (Fig. 4), flying stably for up to 20 minutes.

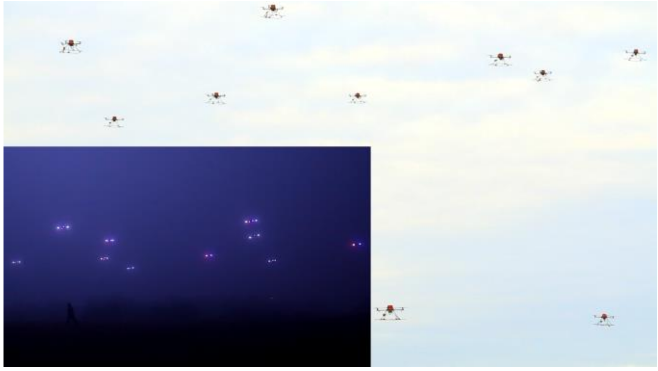

Figure 4. Snapshots of quadcopter flocks of 10 units. Nightlight image: robots are following a target in a grid formation (r_0 = 6 m, v_0 = 3 m/s). Daylight image: SPP model (r_0 = 10 m, v_{flock} = 2 m/s).

We implemented a flock of two hierarchical levels with a leader robot moving along a pre-defined rectangular path and 9 other robots following the leader in a stable, tight grid formation (Fig 5). To assess the coherence of the flock, we calculated the velocity correlation of the robots as a simple order parameter in the range of [-1, 1]:

$$\Phi = \frac{2}{N(N-1)}\sum_{i<j}\frac{\vec{v}^i\vec{v}^j}{|\vec{v}^i||\vec{v}^j|}. \tag{10}$$

The resulting high velocity correlation indicates the stability of the flock (see caption of Fig 5. for details).

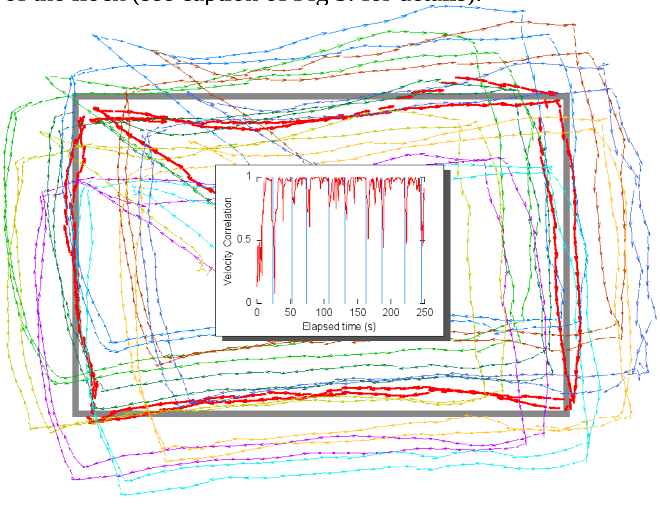

Figure 5. GPS tracklogs of 9 quadcopters in a flock flight, following a 10th leader kopter as a target (red thick arrows). The target takes off from the center and follows a pre-defined rectangular path (grey rectangle) twice. Inset shows the velocity correlation (Φ) of the flock, with $\overline{\Phi} = 0.84 \pm 0.21$ for the t=[25, 250]s period (without take-off). The low-correlation spikes correspond to the direction change at the corners of the path (indicated by blue lines in the inset). The smooth tracklogs and the high velocity correlation are both indications of stable flocking. Average wind

TABLE I. OPTIMAL VALUES OF PARAMETERS

Name	Value	Unit	Comment
pair potential lattice constant (r_0)	6-12	m	Slack is given due to the large GPS position estimation error
pair potential spring constant (D)	1	$1/s^2$	
friction coefficient (C)	10–20	m^2/s	Higher values result in more over-damped, slower but more stable flocks
preferred velocity (v_{flock}, v_0)	2–4	m/s	Note that the max flocking/tracking velocity should be smaller than the max allowed velocity
TRG coefficient (α)	1		A slightly different behavior is achieved if $\alpha+\beta=1$ and there is no need for cut-off
COM coefficient (β)	1		A slightly different behavior is achieved if $\alpha+\beta=1$ and there is no need for cut-off
transfer function deceleration slope (v_0/d)	0.4	1/s	note that this fixed slope is defined on the distance-velocity plot

speed was less than 3 m/s. Waypoint rectangle size is 60x40 m. Other parameters: v_0 = 2 m/s; r_0=10 m.

We successfully implemented the same setup with formation flights of rotating ring and line shapes, as well (Fig. 6). We could change equilibrium distance and switch between shapes in real-time via the R/C remote controller to achieve true self-organization of the formations while adapting to the new situations. General stability is quantified by the standard deviation of the closest neighbor distances of all units, which can be as small as 1 m throughout 1-10 minute periods of a maintained formation (Fig. 6).

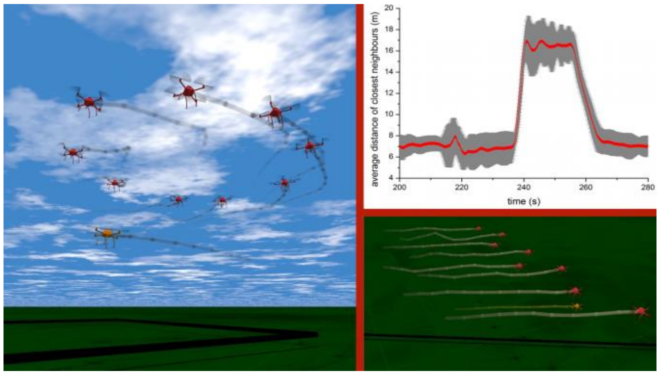

Figure 6. 3D visualization snapshots based on GPS tracklogs of a real target tracking setup with formations of a rotating ring (left) and a line (bottom right). The yellow copter is the leader moving above a pre-defined path (black line), the red ones follow the leader with stable 2D formations. Top right: Average (red curve) and standard deviation (grey fields) of closest neighbor distances in the rotating ring setup. Equilibrium distance is increased from 7m to 17m and back through the R/C remote controller. The small standard deviation (comparable to GPS position error) indicates the stability and smoothness of the formation. $v_{rotation}$ = 2 m/s.

The last setup we tested was a self-propelled model inside a ring shaped area (r_{in}=15 m, r_{out}=45 m). Fig. 7 shows consecutive periods of an emerging self-organized flock of 9 copters circling inside the flight area just like boats [28] or locusts [29] do in a similar setup.

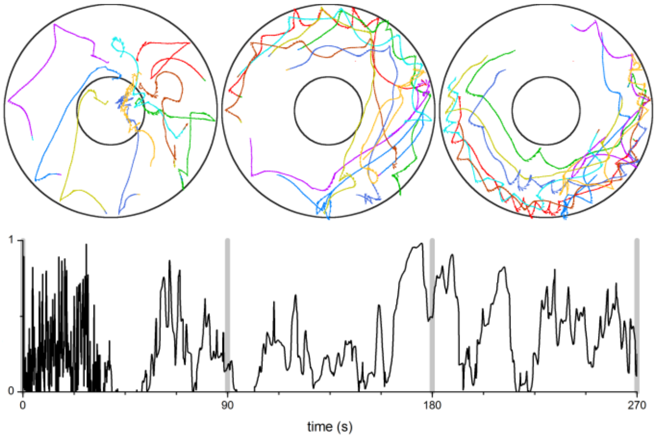

Figure 7. Top: GPS tracklogs from three consecutive 90 s periods of a SPP flock in a circular area after take-off. Bottom: velocity correlation corresponding to the three trajectory plots. Correlation cannot get high for long periods due to the circular area constraint, but the flock tends to organize itself into a coherent swirling motion inside the arena. r_0=10 m; v_0 = 2 m/s.

We performed numerical simulations to investigate the robustness of the system, especially when the characteristic size of the expected flock is larger than the communication range r_C. Stability of flocking behavior can be measured locally with the parameter below:

$$\Phi(\mathrm{r}) = \frac{1}{T}\int_0^T \langle\frac{\vec{v}^i\vec{v}^j}{|\vec{v}^i||\vec{v}^j|}\rangle_{|\vec{r}^{ij}|<r}. \qquad (11)$$

$\Phi(\mathrm{r}) \approx 1$ means that the velocity vectors of units close to each other (closer than r) are parallel during the simulated experiment. According to Fig. 8, stable flocks can be observed with characteristic size greater than $5r_0$, if r_C >3r$_0$.

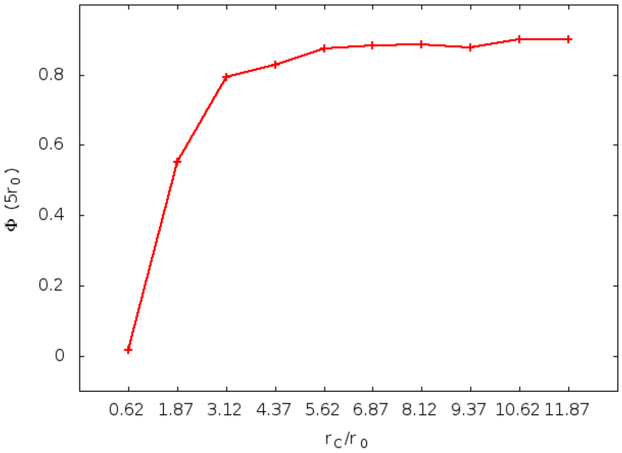

Figure 8. Simulation results about $\Phi(r = 5r_0)$ as a function of communication range r_C. All data points are averaged over a 10 minute simulated experiment with 100 agents, within a 300 m wide square-shaped area and random initial conditions. The environmental effects are modeled with a Gaussian noise term added to the dynamics with $\sigma = 0.1m^2/s^2$. The time delay of the communication is 1s. Other parameters: $C_{frict} = 30\frac{m^2}{s}. r_0 = 8m,\ D = 1s^{-2},\ v_{flock} = 2\frac{m}{s},\ \Delta t = 1s,\ \tau = 1s.$

Conclusions

Decentralization in multi-robotic systems is one promising way of the future. Accordingly, our flocking algorithms run locally on every robot and do not rely on a ground station or any central data processing or control. We demonstrated the usability of our system with widely available GPS-based positioning; however, the framework is general, the algorithms work with any other sensory input from which relative position, velocity and attitude information can be obtained. Moreover, our approach results in a general high-level hardware and software control layer that can be easily transported to other swarming systems, as well.

Stability of the flock fundamentally depends on the sensory errors and the delays in the system. GPS-INS (inertial navigation system) fusion could provide a method for more precise outdoor position and velocity estimation, while a control algorithm with more accurate feed-forward terms or learning capability could reduce the control relaxation time and thus the self-excited oscillations.

Our current algorithms are two and a half dimensional. Flocking and formation flights are performed in two dimensions but units vary their altitude during individual take-off and landing. Group members have the possibility to join or leave the flock any time in the third dimension and they do not have to interact if they have sufficient altitude gap between them. This behavior is similar to most natural aerial flocks, like V-shaped migrating birds, thermalling birds or paragliders. In theory, the flocking algorithm can be extended to three dimensions, but in practice the third, vertical dimension has to be treated quite differently in all cases. Due to gravity and the flight characteristics of quadcopters height control has completely different dynamics compared to position control.

With our current setup we could achieve a minimum of 6–10 m equilibrium distance between the robots in the 0–4 m/s velocity range. Considering ±2 m GPS accuracy, 1.5 s overall time lag and a random disturbance of wind, this is already the closest distance one can allow without the risk of collisions. In theory, the algorithms are not limited to the tested flock density or velocity range. With the dynamically tunable equilibrium distance a wide range of applications and flight situations can be handled. However, tighter flocks require better positioning accuracy, and higher speed implies a loosened flock to allow for sufficient 'breaking distance' between units.

The true advantage of a flocking flight over a single flying robot stands in its increased 'awareness', robustness and redundancy. The flock, as a meta-unit, can detect the environment more efficiently and can operate much longer than its members individually, such as cells in a living organism or migrating locusts over the ocean. The application potential of our system is large, ranging from ad-hoc mobile networks through distributed, self-organized monitoring of the environment (highway traffic, environmentally protected areas or agricultural lands) to stock delivery, rescue operation assistance, pest control, autonomous airport traffic control or even military applications. However, by demonstrating the stable flight of a truly autonomous, decentralized robotic flock, our main goal was to show that the various peaceful applications of drones are by now feasible.

REFERENCES

[1] T. Vicsek and A. Zafeiris, "Collective motion," Physics Reports 517 (2012) 71–140

[2] Dell'Ariccia, G., Dell'Omo, G., Wolfer, D.P., Lipp, H.P., 2008. Flock flying improves pigeons' homing: Gps-track analysis of individual flyers versus small groups. Animal Behaviour 76, 1165–1172.

[3] Bazazi, S., Ioannou, C.C., Simpson, S.J., Sword, G.A., Torney, C.J., Lorch, P.D., Couzin, I.D., 2010. The social context of cannibalism in migratory bands of the mormon cricket. PLoS ONE 5, e15118.

[4] Hoare, D.J., Couzin, I.D., Godin, J.G.J., Krause, J., 2004. Context-dependent group size choice in fish. Animal Behaviour 67, 155–164.

[5] RoboCup homepage: http://www.robocup2013.org/. Accessed: February, 2014

[6] Mas, I., & Kitts, C. (2012). Object Manipulation Using Cooperative Mobile Multi-Robot Systems. In Proceedings of the World Congress on Engineering and Computer Science (Vol. 1).

[7] Turgut, A. E., Çelikkanat, H., Gökçe, F., & Şahin, E. (2008). Self-organized flocking in mobile robot swarms. Swarm Intelligence, 2(2-4), 97-120.

[8] Murata, S., & Kurokawa, H. (2012). Self-organizing robots. Springer.

[9] Rubenstein, M., Ahler, C., & Nagpal, R. (2012, May). Kilobot: A low cost scalable robot system for collective behaviors. In Robotics and Automation (ICRA), 2012 IEEE International Conference on (pp. 3293-3298). IEEE.

[10] Kernbach, S., & Kernbach, O. (2011). Collective energy homeostasis in a large-scale microrobotic swarm. Robotics and Autonomous Systems, 59(12), 1090-1101.

[11] Welsby, J., Melhuish, C., Lane, C., & Qy, B. (2001). Autonomous minimalist following in three dimensions: a study with small-scale dirigibles. Proceedings of Towards Intelligent Mobile Robots Manchster.

[12] De Nardi, R., & Holland, O. (2006). Swarmav: A swarm of miniature aerial vehicles. In Proceedings of the 21st Bristol International UAV Systems Conference, April 2006

[13] Hauert, S., Leven, S., Varga, M., Ruini, F., Cangelosi, A., Zufferey, J. C., & Floreano, D. (2011, September). Reynolds flocking in reality with fixed-wing robots: communication range vs. maximum turning rate. In Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on (pp. 5015-5020). IEEE.

[14] Cs. Virágh, G. Vásárhelyi, N. Tarcai, T. Szörényi, G. Somorjai, T. Nepusz, T. Vicsek (2014) Flocking algorithm for autonomous flying robots. Submitted to Bioinspiration and Biomimetics, special issue on Bioinspired Flight Control

[15] Reynolds, Craig W. (1987) Flocks, herds and schools: A distributed behavioral model, ACM SIGGRAPH Computer Graphics, 21(4), pp. 25-34

[16] Stirling, T., Roberts, J., Zufferey, J., & Floreano, D. (2012, May). Indoor navigation with a swarm of flying robots. In Robotics and Automation (ICRA), 2012 IEEE International Conference on (pp. 4641-4647). IEEE.

[17] Kushleyev, A., Kumar, V., & Mellinger, D. (2012, July). Towards A Swarm of Agile Micro Quadrotors. In Robotics: Science and Systems.

[18] Bürkle, A., Segor, F., & Kollmann, M. (2011). Towards autonomous micro uav swarms. Journal of intelligent & robotic systems, 61(1-4), 339-353.

[19] Homepage of Mikrokopter Co., Germany: http://mikrokopter.de. Accessed: February, 2014

[20] Vicsek, T., Czirók, A., Ben-Jacob, E., Cohen, I., & Shochet, O. (1995). Novel type of phase transition in a system of self-driven particles. Physical Review Letters, 75(6), 1226-1229.

[21] Orosz, G., Wilson, R. E., & Stépán, G. (2010). Traffic jams: dynamics and control. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 368(1928), 4455-4479.

[22] Koren, Y., & Borenstein, J. (1991, April). Potential field methods and their inherent limitations for mobile robot navigation. In Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on (pp. 1398-1404). IEEE.

[23] Helbing, Dirk, Illés Farkas, and Tamas Vicsek. "Simulating dynamical features of escape panic." Nature 407.6803 (2000): 487-490.

[24] Cucker, F., & Smale, S. (2007). Emergent behavior in flocks. Automatic Control, IEEE Transactions on, 52(5), 852-862.

[25] Han, J., Li, M., & Guo, L. (2006). Soft control on collective behavior of a group of autonomous agents by a shill agent. Journal of Systems Science and Complexity, 19(1), 54-62.

[26] Webpage for the best known packings of equal circles in a circle: http://hydra.nat.uni-magdeburg.de/packing/cci/cci.html#cci1. Accessed: February, 2014

[27] Helbing, Dirk, Illés Farkas, and Tamas Vicsek. "Simulating dynamical features of escape panic." Nature 407.6803 (2000): 487-490.

[28] N. Tarcai, Cs. Virágh, D. Ábel, M. Nagy, P. L. Várkonyi, G. Vásárhelyi and T. Vicsek (2011) Patterns, transitions and the role of leaders in the collective dynamics of a simple robotic flock Journal of Statistical Mechanics: Theory and Experiment 2011 P04010

[29] Yates C. A. et al., Inherent noise can facilitate coherence in collective swarm motion, 2009 Proc. Nat. Acad. Sci. 106 5464