DOI 10.12694/scpe.v16i3.1099
Scalable Computing: Practice and Experience ISSN 1895-1767
Volume 16, Number 3, pp. 233–247. http://www.scpe.org

APPLICATION OF COLLECTIVE MOVEMENT IN REAL LIFE SCENARIOS: OVERVIEW OF CURRENT FLOCKING SOLUTIONS*

BERNÁT WIANDT †‡ ANDRÁS KŐKUTI †§ AND VILMOS SIMON †¶

Abstract. Today more and more attention is given to collective movement of dynamic nodes, called "flocking". The main problem is how to control the nodes to behave as a group and make them move together to destinations (and moreover complete some tasks such as sensing, transporting, etc.). Currently there are many works, which attempt to solve the flocking problem, but very few of them are suitable for real world use. This is because most of the algorithms are overlooking crucial physical limitations or the used model is not detailed enough. In this paper we survey the existing flocking algorithms, including a review of the most important requirements for real application scenarios.

Key words: collective movement, flocking, self-propelled particles, self-organising

AMS subject classifications. 68M, 68U

1. Introduction. Collective motion, which is one of the manifestations of a more general class of phenomena, called collective behaviour, has attracted many researchers from diverse fields of scientific and engineering disciplines. From the area of biology, the grouping of animals is a natural phenomenon, in which a number of animals are involved in joint movement and formation of a group. Examples [35, 37, 31] are insect swarms, fish schools, herds of wildebeests, etc. The main goal of these works is to figure out a model of the grouping, with respect to the dynamics of animal motion. A more general phenomena of collective behaviour can be observed in atoms and molecules as well, for example, during spontaneous magnetisation [29]. Moreover, the concept of collective action exists not only in animals but also in human societies [15]. The flocking phenomena (the notion of flocking is used as a synonym of collective motion) can be an appropriate solution in many engineering applications as well. Applications of flocking include massive mobile sensing in an environment; parallel and simultaneous transportation of vehicles or delivery of payloads; and performing military missions, such as reconnaissance, surveillance, and combat using a cooperative group of Unmanned Aerial Vehicles (UAVs). A flocking group of robots can perform tasks like exploration of an area [24], autonomous navigation for deployment, surveillance or search and rescue operations. Aircraft flying in formation can use the available fuel supply more efficiently. When the formation is given the flexibility to be able to reconfigure itself, the error tolerance and value of the ensemble increases as it will be capable of performing tasks in more diverse environments. Formation reconfiguration enables the vehicles to mitigate certain errors or faults, such as sensor or actuator failures, failure of a complete aircraft and communication problems.

The first work [40] in this area was a computer graphics animation of a group of birds by Reynolds. His bird-like objects, called "boids", moved along trajectories determined by differential equations, taking into account three types of interactions:

Flock Centering Attempt to stay close to nearby boids.
Collision Avoidance Avoid collisions with nearby boids.
Velocity Matching Attempt to match the velocity of nearby boids.

The rules mentioned above are also known as *cohesion*, *separation*, and *alignment* in the literature. These three forces are the core of all the models in the surveyed literature, however, the implementation of them always differs and depends heavily on the used model. Additions to the basic three forces may appear in some works, for example Vicsek et al. in [52] propose an additional fourth force dampening the control inputs and therefore making the flock more stable.

*This paper has been partially supported by HSN Lab, Budapest University of Technology and Economics, http://www.hsnlab.hu
†Budapest University of Technology and Economics, Department of Networked Systems and Services
‡bwiandt@hit.bme.hu
§kokuti@hit.bme.hu
¶svilmos@hit.bme.hu

Aside from the common rules, there are some common paradigms in all algorithms. All reviewed implementations use the self-propelled particle model. In this model, the states (position, velocity, acceleration) of two individual units change during their interaction. On the other hand, in equilibrium systems the total momentum is preserved and that is how the velocity distribution is being built up. There are three control methods, which are used widely in the literature. Some of the algorithms are *unit-centre referenced*, where only local information is used (information gathered from the node itself), therefore nodes are independent in the ensemble, while others depend on information collected from neighbours, this is referred to as *neighbour-referenced*, but most of them suit the *leader-follower* model, where there is one or more leader(s) of the whole group commanding other nodes.

In contrast to the huge number of flocking algorithms already published in the literature, in our opinion, very few of them are capable of real world use. There are many requirements (such as obstacle avoidance, scalability, tolerance to communication link errors, dealing with noisy measurements and time delays, etc.), that have to be satisfied by an algorithm to succeed in achieving flocking in a real environment. These requirements are usually only partially satisfied by the surveyed algorithms, as for example in a formal proof about the consistency of the flock, the communication errors (such as packet loss) are not taken into account, or in a simulation framework, the real physical conditions cannot be fully simulated.

The rest of the paper is organised as follows: in Sec. 2 we give a short description of our preferences when examining the proposed flocking solutions along with some introduction to the basic ideas involved in the field. In Sec. 3 we give a comprehensive overview of the control schemes involved in flocking systems and in Sec. 4 we categorise the models used in the reviewed articles. In Sec. 5 we review the communication challenges and solutions and in Sec. 6 we describe some of the simulators and demonstrations connected to flocking.

2. The examined properties of flocking solutions. The main focus of our analysis in the following sections revolve around the applicability of published flocking algorithms in real life. We do not have the resources to build the algorithms into real hardware and test them in a variety of environments, therefore we approach this complicated question by categorising the available works based on how elaborate is the used model and trying to find some kind of evolution article by article.

When we talk about flocking and flocking algorithms, the properties of the environment (i.e., the used model) is one of the most important aspects. The restrictions imposed by the environment have effects on the mathematical formulation of the algorithm, the possible communication and movement models, and it defines the strength of the formal proofs given by the authors. There are numerous questions arising when talking about a physical model, such as the boundedness of the control output, the needed communication bandwidth, the available information, the used vehicle model, etc. For example, when trying to avoid collisions, some articles assume an unbounded control output is possible, which is clearly not the case in a real life scenario. Designing precise control for a complicated vehicle involves a much more elaborate physical model, for example one would have to take into account the intrinsic nonlinearities, the external disturbances influencing the vehicle, the varying weight distribution caused by burning the available fuel or by picking up a payload and the inherent noise and imprecisions in the sensors and actuators.

Flocking algorithms usually depend on some kind of information gained through communication. By nature of the nodes involved, the communication is always wireless, which poses another interesting problem. How to model the connection between the nodes in order to be realistic and also try to keep the model as simple as possible? Flocking algorithms usually do not pay attention to communication aspects, such as the wireless technology used, modulation, interference, frequency band, packet loss, wireless link strength, available bandwidth and so on. Most of the models assume a simple graph based communication network, where two nodes with positions q_i and q_j can communicate if $\|q_i - q_j\| < r$, r being the interaction range. The notion of neighbourhood is introduced to represent the nodes in the 2D circle or 3D ball with radius r around the node. This definition of neighbourhood may not be the one that is actually used in bird flocks. In [9] authors conclude, that bird flocks (starlings in this case) are using a topological neighbourhood rule, where the number of neighbours are fixed and selection of neighbours are independent of their physical distance. The simple communication model used in flocking algorithms has a number of different forms, starting with the *undirected*, where we assume that if node i can communicate with node j, then it is true the other way around. In real wireless systems this is not always the case as there can be simultaneous radio transmissions going on around node i and j. *Directed*

communication models incorporate this effect, but are harder to reason about. Connectivity, the available communication links and their quality, is usually captured with an adjacency matrix or some kind of extended form of it. Several works discuss the connectedness of the flock and deliver mathematical proofs upon the preservation of connectivity through time. New connections in this model are formed if two nodes are close enough to each other, however this poses a problem, when the distance between them end up around r: the connections between them can form and be destroyed rapidly. This can be remedied by introducing some kind of hysteresis when adding new neighbours [53] and is used by some works that recognised this problem.

Information about the node itself and especially about its neighbours are not always available and also not accurate. If the information needed in an algorithm can be reduced, simpler hardware can be used and fewer assumptions are required in order for the algorithm to fulfill its duty. Some works require the position and velocity information to be available and accurate in order to achieve flocking. Others assume, that the position information is noisy and only the relative distances can be measured accurately. Some depend on a reduced set of available data and define certain tradeoffs, for example, in the second order case (double integrator dynamics) require only the position and velocity information but from the 2-hop neighbours too, or require only the heading angle to achieve flocking but introduce a gain factor in the control equation, which has to be "large enough". In this case this simplifies the whole system, but introduces larger oscillations between nodes due to the large control gains. These oscillations can result in a quite erratic behaviour of the flock and eventually can cause the flock to disperse. In [52] the authors recognised this problem and proposed a damping factor in the control output to prevent such oscillations. They use the same approach in the implemented control algorithm on their drones [51], a real world, truly autonomous flocking implementation of UAVs.

Simulations are usually done to prove the claimed properties in one or more trials, but simulators can contain bugs, or simulations can be set up to show the best case scenario for a proposed method. Also simulators suffer from the same problems as described in the beginning of this section, namely the simplistic assumptions about the real world, mobility models and communication. Therefore the ultimate proof of a mathematical formula or a proposed algorithm is the real world trial, where the authors take the effort of implementing the solution to actual physical nodes and run, measure and evaluate performance characteristics of their work. We do know that all of these requirements is impossible to meet in one publication but we strive for these and will try to show the reader a way among the vast amount of work done in this field up to date.

3. Control mechanisms for flocking. A very important property of a flocking algorithm is the control mechanism, which is responsible for directing the flock to the desired destination and managing the shape of the group. Various mechanisms have been developed for different use cases to be introduced in this section.

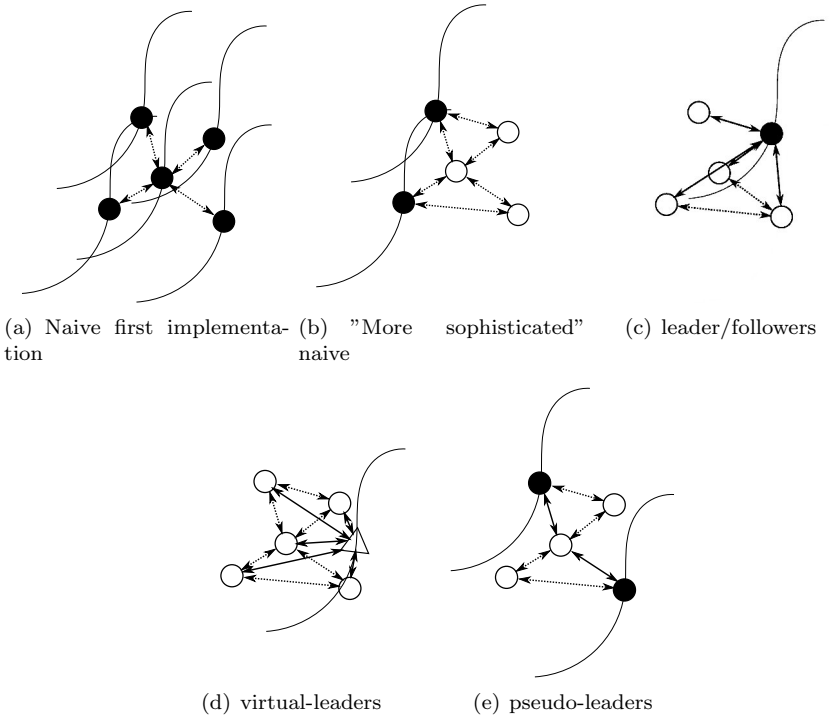
The simplest implementation is if each node knows the trajectory or the previously determined direction [40], in which case no control overhead is needed. However, in a real life scenario the trajectories or even the directions are not previously determined and sometimes both parameters are changing from time to time (in a real flock it is essential to change the destination on the fly, for example in a search and rescue operation).

A more sophisticated mechanism is proposed in [13], where the nodes can be separated into two distinct sets. In the first one each individual is "naive", which means it does not possess any kind of trajectory or directional information, thus nodes in the first set can only follow the flock. The nodes in the second set are "informed", hence they know the desired trajectory or direction. In this model, the second set of nodes do not indicate that they are "informed", but instead guide the rest of the swarm by moving in the preferred direction. With this mechanism the user manages only a few of the interacting nodes, but due to the used flocking model, they can influence each participant in the swarm. Findings and results from several references show that a self-organised flock of nodes can be effectively guided by a minority of informed nodes within the flock. However, a huge shortcoming of this solution is that the swarm will only slowly converge to the preferred direction, since the "informed" nodes are not turning instantly, instead the whole flock will turn to some weighted sum of the current and preferred direction, and in the next time slot they will turn more closely to the desired direction and so forth, which adds a rather large delay to a target tracking application.

To overcome the previously described problem, the *leader/followers* [33] solution can be employed, which is a widely used control mechanism in the literature [34, 1]. In this control scheme a special role is assigned to a node (called the "leader") to guide the whole flock (and perhaps to maintain the formation). The main difference from the two distinct set scheme is that other individuals (called "followers") are following the leader,

thus they know the existence of a leader in the flock who should be followed. A serious disadvantage of this type of control mechanism is that it does not scale well with an increasing number of nodes since the position error propagates cumulatively by disseminating the information from the leader to followers, and that all nodes have to communicate (directly or indirectly) with the leader, while the wireless radio ranges are usually bounded.

The enhanced version of the previous solution is the *virtual-leader* [25] or *virtual-leader/followers* [36] mechanism. Contrary to the *leader-followers* scheme, where the leader is one of the physical agents (e.g., neighbouring a vehicle in a multi-vehicle system or a fish in a school), here it is only a virtual reference point (beacon), that can influence the neighbouring nodes. In this model a new force is introduced, guiding the nodes, similar to the inter-node forces, except that it defines the force on a physical node in reference to a virtual leader. Since there is no physical node with a special role (such as the leader in the previous scheme), all of the nodes are interchangeable with each other. The main benefit of this approach is the robustness of the group to the failure of an individual agent. Because of its beneficial properties, it can be found in many flocking solutions [56, 19, 49, 53].



(a) Naive first implementation (b) "More sophisticated" naive (c) leader/followers

(d) virtual-leaders (e) pseudo-leaders

FIG. 3.1. *A simple example for each of the described mechanisms. The agents are represented with circles (the filled ones possess information about the desired trajectory / direction) the triangle indicates a virtual agent (or a reference beacon), while the arrows signal the inter-node forces (potentials). The dashed arrows present a usual interaction (such as cohesion, separation, etc.) while the solid ones indicate the controlling interactions. Controlling interaction is used only in cases when a follower node is informed about a leader to be followed.*

The extension of the *virtual-leader/follower* mechanism is the *pseudo-leaders* [65] model. This model is very similar to the previous one except that it is not essential for all agents to be informed [43], and a pseudo-leader represents an agent, while in the previous model there were just reference beacons. A flaw of this solution is that the selection of pseudo leaders [65] works only on a fixed topology, while in a real life scenario, in the majority of the cases, the topology varies dynamically.

There are other schemes [45, 4], where no leaders are designated and nodes react to the positions of their neighboring nodes, while moving in an emergent direction, but their impact to the flocking algorithms is less compared to the previous models.

For a better understanding, the described mechanisms are depicted in Fig. 3.1. This figure does not show

the whole picture, since not all inter-node interactions (depicted with arrows) are illustrated. The controlling mechanisms differ from each other in how many leaders exist in the flock, how they control other nodes, and how the other nodes are informed via communication links. As it can be concluded from this section, constructing an appropriate controlling mechanism to be feasible in real life is a huge and complex task, therefore some widely used patterns are the basis of almost every flocking algorithm.

4. Physical models for flocking algorithms. The physical model assumed in any work constrains the control algorithm, influences the capabilities of the system and is one of the most important factors when designing the control algorithm. The model describes the dynamics of the flock members, the interactions between them, possible environmental noises and external disturbances. Let us enumerate the different physical models involved in the reviewed research, starting from the simplest to more detailed and complicated. The models we examined are based on the assumption, that the vehicle or flock member is particle-like and it can propel itself in the direction and with the velocity desired, based on the used control algorithm. Usually the size of the vehicle is not considered in the models, rather it is accounted for when designing the collision avoidance part of the control algorithm, by enforcing a constraint on the minimal distance between two vehicles. The simplest particle-like model is a linear-integrator type system, the simple integrator, also known as first-order kinematics, defined by Eq. 4.1.

$$\dot{q}_i = u_i, \qquad i = 1...n \text{ and } q_i, u_i \in R^m \text{ (e.g., } m = 2, 3) \tag{4.1}$$

In first order kinematics, the control input is the velocity vector of the vehicle. This is the easiest model, it is far removed from the real world but is easy to reason about. It is used as a preliminary analysis tool in [20, 60, 57]. A far more usable and widespread model is the double integrator, a.k.a. second-order kinematics. The model is defined by Eq. 4.2.

$$\dot{q}_i = p_i \text{ and } \dot{p}_i = u_i \qquad q_i, p_i, u_i \in R^m \text{ (e.g., } m = 2, 3) \tag{4.2}$$

being the position and velocity of the node i respectively, and u_i is the control input for node i. A basic assumption is that the node can be controlled in any direction, it does simple integration, it has no weight or other interfering forces from the outside world. The control algorithm is designed to generate u_i, based on the input variables, such as the current position, velocity and acceleration of node i and some subset of the same information from its neighbours. Information flow between nodes is crucial, as control algorithms need input data to generate the correct output, therefore move the node in the desired direction. One of the used descriptions of communication is the unit-disc model, where nodes within a specified distance in space can communicate with each other. These models are usually undirected, meaning that for all participants: if node i can communicate with node j, then node j can communicate with node i. This model is prevalent and used for example in the landmark paper of Olfati-Saber [36], where they study the possibilities of flocking with a virtual leader-follower type model. The solution for the flocking problem in [36] is based on virtual potential functions, such as the three basic forces (cohesion, separation, alignment) is modelled with potential functions either attracting or repelling other nodes.

The Olfati-Saber approach to flocking assumes that information about the virtual leader is available to all flock members, which is clearly not practical, nor scalable. Subsequent works in the field proved that flocking can be achieved, when only a (small) fraction of the nodes are informed about the virtual leader's state [47, 48, 43]. It is a beneficial property, because this way the flock needs less communication and the uninformed members can be driven by the informed ones. Although in [43] it is proven, that a subset of nodes is enough to be notified of the virtual leader's state, the selection algorithm was not provided and a clear rule as to which nodes should be informed was not given. In [21], authors give a criterion to choose the nodes to be pseudo leaders, therefore they provide a tool to actually implement the previous approach. Pseudo leaders in this context mean the nodes informed about the virtual leader's state. Virtual leaders can be static or dynamic: in the static case their velocity vector is static and in the dynamic case it can be time-varying. Dynamic virtual leaders opened the possibility to target tracking applications with flocks and enabled the free movement of the flock in space.

Dynamic virtual leaders were investigated in [48], which is an extension to the classic Olfati-Saber approach, and is an assumption for almost all reviewed works.

The information required to achieve flocking is limited in [10], as in the first-order case the velocity measurements are not required and in the second-order case accurate acceleration measurements of the leader and followers are not required. Reducing input to the control algorithm is a desirable property, because accurately measuring acceleration and velocity and transmitting it to the neighbours in a timely fashion is not easy to implement, if at all possible, without considerable time-delays and/or inaccuracies. There could be hardware or budget limitations, which make it impossible to use that information. Limiting the input variables of the flocking algorithm makes it easier to implement it and makes it more robust, because higher order information (velocity compared to acceleration and position compared to velocity) can be measured more accurately and with simpler hardware. There has been a lot of effort dedicated to this subproblem, for example [61] for flocking, using only sampled position data or [66] for attitude coordination with the same constraints or [64], which deals with the case, where the flock consists of heterogeneous agents, meaning their movement dynamics and constraints are not the same.

Physical implementation of the controllers on each node has some limitations. Usually this manifests in the boundedness of the control input, that can be satisfied by the controller. This means that we should take this effect into account when developing the control algorithms to achieve flocking using physical hardware. Two of the earliest works dealing with this subproblem are [28], researching flocking in general, and [22], which details an algorithm for flocking with a virtual leader with time-varying velocity. There was some research conducted in the direction of implementing velocity-free bounded controllers, but early works were concentrating only on the velocity consensus or were assuming a fixed interaction topology [64, 2, 3]. Recently, in [16], authors introduced a velocity-free and bounded controller design.

Another interesting subproblem is obstacle avoidance, where the problem is twofold: on the one hand we have to ensure that the nodes will not crash into each other, on the other hand we somehow have to avoid crashing into some bigger obstacle, like a wall. The first half of the problem (usually referred to as collision avoidance) is usually solved by introducing some requirement on the minimal distance between nodes, and ensuring that they will be met at all times by factoring this requirement into the designed virtual potential functions, where the dimensions of the used vehicles can be accounted for. The second half of the problem, avoiding collision with bigger, static or dynamic obstacles was addressed in [36], where the obstacles were modelled as a virtual node in the system. In that work, artificial potential functions (APFs) were used to calculate the control output and the obstacle nodes were designed to allow the nodes closing by to move past them in an efficient way. An interesting approach was proposed in [54], where the authors argue, that APFs have the problem of local extrema (multiple obstacles in an unfortunate configuration, stuck nodes, etc.) and moving around an obstacle will sometimes fail. In their work, stream functions were used to calculate the paths of nodes around obstacles, ensuring stable flocking and connectivity maintenance during obstacle avoidance.

A straightforward extension of the second-order model is to incorporate the intrinsic nonlinearity of the vehicle into the model and move a bit away from the ideal world. The equations for the general case are defined in Eq. 4.3.

$$\dot{q_i} = p_i \text{ and } \dot{p_i} = f(p_i, q_i) + u_i, \text{ where } q_i, p_i, u_i \in R^m \text{ (e.g., } m = 2, 3) \tag{4.3}$$

$f(p_i, q_i)$ represents the intrinsic nonlinear characteristics of the nodes. Recently, nonlinear models got a lot of attention, as they are one step closer to reality. In earlier works dealing with consensus problems (matching velocities among flock members), the authors studied first- and second-order systems with intrinsic nonlinear characteristics, using pseudo-leader [65] or virtual leader [19] approaches. In [65, 19] the nonlinear term in the model is only dependent on the position of the node, however a more realistic model would also depend on the velocities of the nodes. Also, a globally Lipschitz-like condition is generally used in order to formulate appropriate control, and prove that all the agents can synchronise with the leader(s) in [65, 19, 58] and in the second order case in [59, 46]. Based on those two observations, Su et al. proposed an adaptive control algorithm in [49] for flocks with virtual leaders, where the nonlinearity is only locally Lipschitz and the nonlinearity terms in the model depend both on the velocity and position of the nodes, therefore the model used in their article is more general than the previous approaches.

Previously we stated, that second-order systems are the most prevalent and in [10] it was proved, that for simple second order systems there is no need to have all the available information about the neighbours and virtual leader(s) to achieve flocking. In [53] the authors argue, that nodes might be governed by nonlinear dynamics and nonlinear dynamics is commonly used in synchronisation of complex dynamical networks [55, 26, 58]. In [18] the model is a bit more evolved as measurement uncertainties and external disturbances are accounted for and a neural network approach is proposed to achieve consensus with first- and second-order dynamics. This algorithm is dealing only with achieving consensus and rendezvous in a prescribed place. The work carried out in [14] extends this by solving the leader-follower problem, allowing the leader to have a time-varying state trajectory. Up to this point we only talked about agents with uniform intrinsic dynamics, but in real world applications heterogeneous dynamics should be accounted for. Wang et al. proposed an approach for this exact situation in [53], allowing for heterogeneous nonlinear dynamics for the leader and the followers, implementing collision avoidance by defining APFs and introducing a hysteresis-like method for acquiring new neighbours.

Another model proposed, is the Standard Vicsek Model for self-propelling particles, which in it's simplest form is a cellular automaton like approach to describe the motion of self-propelled particles. In this model each node updates its velocity vector by averaging their neighbours' heading angles and adding some random perturbations, to model the various factors influencing the motion of the nodes.

$$v_i(t+1) = v_0 \frac{\langle v_j(t) \rangle_R}{|\langle v_j(t) \rangle_R|} + perturbation \tag{4.4}$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{4.5}$$

Equations 4.4 and 4.5 define the model, where $\langle ... \rangle_R$ denotes the average of the nodes' velocities in a circle of radius R. Perturbations in the simplest case are random angles added to the average calculated in $v_i(t+1)$. In this model if some nodes decide to turn, their turning information will be damped by their neighbours' averaging of the heading angles and this will result in a slow and uncoordinated turning motion. In [12], authors recognised that in order to reconstruct real flocking motion while turning, there needed to be a conservation of rotational symmetry implemented in the system and there needs to be a behavioural inertia mediating the effect of the social force. Therefore they propose a new model, from which the Standard Vicsek Model can be obtained as the over-damped limit of the system. Members of the flock derive their velocity vector from their neighbours and the selection of neighbours can have a huge effect on the stability and structure of the flock. In the simplest case, neighbours are the nodes around us and closer than a predetermined distance, usually called the interaction range. In 2D this defines a circle, in 3D a ball around each node. However, real flocks (for example bird flocks) do not function that way. They select the neighbours from which they derive information in a topological manner. In [9] authors test the hypothesis, that proper topological neighbour selection is beneficial, and conclude that compared to the metrical neighbour selection scheme (circle or ball around the nodes), it resulted in a more stable and more cohesive structure. Topological interaction means that each node interacts with a fixed number of other nodes, irrespective of their distance. Implementing this with real hardware of course has its limitations, because of the limited range of wireless communication.

The models we investigated are trying to resemble the real world, with increasing complexity, external disturbances and nonlinear intrinsic dynamics. A common shortcoming of these algorithms, mentioned in [23], is that the vehicles in the flock or formation may have additional constraints. For example, the more and more popular UAVs involved in formation flight cannot endure negative velocities if they are of the fixed wing type, therefore the control should be designed with this constraint in mind. A particular subproblem investigated in the article is turning: if the leader of the formation turns too fast, some vehicles should decelerate in order to maintain their position and this consequently limits the turning angle of the leader.

Authors in [39] demonstrated the benefits of using multiple UAVs in a target tracking scenario. In their application, the target identification was implemented by image recognition and this resulted in more constraints on the model, for example the turning speed of each vehicle was maximised, because fast enough turning would

be detrimental in the performance of the image recognition algorithm. Their physical model is highly detailed, incorporates the gravitational force, environmental disturbances (i.e., wind), and is chosen to be a 4 degree of freedom stochastic model. The stochasticity in the system should capture forces missing from the model and environmental disturbances as well.

In Table 4.1 we summarised the properties of some of the solutions mentioned in this section. In the first column there is a symbolic name, usually the last name of the first author and the reference number. The second column titled **Control** is the type of control scheme that was applied in the algorithm. Empty cells mean that there is no explicit control, and the problem solved is a consensus problem, meaning that the goal is to synchronise the velocity vectors of the nodes in the flock. *Virtual leader* means that there is a (virtual) node to be followed by the others in the flock. *Constant* means constant velocity and *dynamic* means variable velocity for the virtual leader. The third column is titled **Informed** and it contains the number of nodes that has to be informed of the leader's state. Consensus problems do not have a leader, so the cell's value is *N/A*, other solutions contain either *all* if all the nodes have to be informed or > 1 if informing at least one node is enough for the flock to achieve its goal. The fourth column is **Linear**, *yes* if the solution assumes linear dynamics for the nodes and *no* if the nodes have intrinsic nonlinear characteristics. **Noise** column is *no* if the algorithm disregards internal uncertainties and external disturbances, *yes* otherwise. **Input** enumerates the needed information about the neighbours in the system. In two cases ([18] and [14]) there is no exact input variable mentioned in the article, just a general state vector is exchanged possibly containing the position, velocity, temperature, etc. Seventh column is called **Heterogeneity** and it distinguishes whether the algorithm considers homogeneous nodes or different dynamics and properties are allowed for the nodes that make up the flock. *No* signifies the homogeneous case, while *yes* means a heterogeneous flock. Collision avoidance is always implemented in the reviewed works, while obstacle avoidance is not. The column titled **Obstacle avoidance** contains a *yes* if there is some sort of mechanism helping the flock overcome arbitrary sized and shaped obstacles in the space, and *no* means that the authors assumed free flocking, meaning there are no obstacles in the field. Lastly **Communication** gives an overview of the properties of the communication assumptions. *varying* stands for a variable neighbour set, *fixed* for a fixed one. *unit-disc* means a unit-disc model, while *undirected* or *directed* distinguishes between the connection types in the communication graph describing the system.

5. Communication between the nodes. Research papers in this field fall mainly into two categories. The first group is composed of solutions that rely on the existence of sensors dedicated to measure the distances between nodes, such as infra-red and/or ultra-sonic sensors [34, 1, 42] or the infra-red short range sensing systems [13]. The second group of solutions are significantly different from the first, since nodes are only measuring their own location (via a GPS receiver) and periodically broadcasting this location information via their radio interface. Solutions from the first approach do not depend on the usage of the radio, however, they work only when nodes are relatively close to each other. Some of these solutions are already used in use cases when smaller areas need to be covered with less mobile nodes, e.g., robots on the ground. The second group is targeting highly mobile and dynamically reshaping flock of nodes, e.g., UAVs, especially for the scenarios when the nodes can autonomously split and rejoin. Algorithms from the second group must solve the problems arising from wireless communication, such as medium usage, asynchrony, packet losses, etc. In this section the communication models and methods from the second category will be described, currently in use by well-known flocking algorithms.

The simplest model is an all-to-all broadcast, meaning that any node can communicate with any other node. In this ideal case the radio range is infinite, the communication is undirected and there are no packet losses assumed. When having such assumptions, there is no need for a sophisticated broadcast method, therefore blind flooding [27] can be applied. Of course this model does not meet any requirement of a real application, but it is used widely [12] for an initial model, since the first priority of the algorithms is to work under ideal conditions.

A bit more realistic [63, 36] case employs a finite radio range. This is an initial model as well and the use of a sophisticated communication protocol is pointless as well, however the communication between nodes is bounded with a finite range (defining a circle in 2D and a ball in 3D around the node). Due to this limitation, the flocking algorithm is more complex in the case of a leader-follower control scheme, as the interaction range is limited, but the current state of the leader has to be forwarded to every node in the flock. This can cause all

TABLE 4.1
Summary of a few selected solutions

Solution	Control	Informed	Linear	Noise	Input	Heterogen nodes	Obstacle avoidance	Communication
Olfati-Saber's [36]	Virtual leader (constant)	all	yes	no	position, velocity	no	yes	varying, unit-disc, undirected
Su's [47]	Virtual leader (constant)	> 1	yes	no	position, velocity	no	yes	varying, unit-disc, undirected
Su's [48]	Virtual leader (dynamic)	all	yes	no	position, velocity	no	yes	varying, unit-disc, undirected
Shi's [44]	Virtual leader (dynamic)	> 1	yes	yes	position, velocity	no	no	varying, undirected
Jin's [21]	Virtual leader (dynamic)	> 1	no	no	position, velocity	no	no	varying, undirected
Cao's [10]	Virtual leader (dynamic)	> 1	yes	no	position, velocity	no	no	varying, undirected
Yu's [61]	-	N/A	yes	no	position	no	no	varying, directed
Zou's [66]	Virtual leader (dynamic)	> 1	no	no	position	yes	no	varying, undirected
Housheng's [19]	Virtual leader (dynamic)	> 1	no	no	position, velocity	no	no	varying, unit-disc, undirected
Yu's [58]	Virtual leader (dynamic)	> 1	no	no	position, velocity	no	no	varying, unit-disc, undirected
Yu's [59]	-	N/A	no	no	position, velocity	no	no	varying, directed
Su's [46]	Virtual leader (dynamic)	> 1	no	no	position, velocity	no	no	varying, unit-disc, undirected
Su's [49]	Virtual leader (dynamic)	> 1	no	no	position, velocity	no	no	varying, unit-disc, undirected
Wang's [53]	Virtual leader (dynamic)	> 1	no	no	position, velocity	yes	no	varying, unit-disc, undirected
Hou's [18]	-	N/A	no	yes	state vector	yes	no	fixed, undirected
Cheng's [14]	Virtual leader (dynamic)	> 1	no	yes	state vector	yes	no	fixed, directed
Wang's [54]	Fixed point	N/A	yes	no	position, velocity	no	no	varying, unit-disc, undirected

kinds of problems, for example increasing error in the perceived position of the leader as we move farther away from it, since there could be nodes, which are not connected directly to the leader and information has to be propagated to them in a hop by hop manner, resulting in large latencies, therefore inaccurate information.

In the previous models the communication graph (where vertices are nodes and edges indicate the communication links among them) was undirected, moreover, in the simplest model the whole graph was complete,

TABLE 5.1
Comparison of the network model being used in flocking systems

Solution	**Active or passive**	**Transferred data**	**Transmission range**
Shi's [42]	passive	N/A	N/A
Navarro's [34, 1]	passive, active	velocity and weights	3.3m
Çelikkanat's [13]	passive, active	heading vector	$\sim$ 20m
Cavagna's [12]	active	position	∞
Hai-Tao's [63]	active	position	r, a distance limit
Olfati-Saber's [36]	active	position	r, a distance limit
Zhang's [62]	active	position, velocity	r, a distance limit
Carlési's [11]	active	velocity and control variable	∞

and in the second case it is composed of many smaller (the size is determined by the interaction range of the device) complete graphs. A realistic communication environment cannot be described as an undirected graph, instead some works have adopted a directed graph, since the transmission ranges can differ from node to node. It is a valid extension, based on the assumptions, that nodes of the flock could be different devices (such as in [64, 62]) or they can vary the power of their wireless transmitter for energy efficiency reasons or the different number of ongoing transmissions can cause varying interference around them. However, neither the medium access, nor the packet collisions are taken into account in this model, hence the naive blind flooding protocol can be used in this case as well.

In a more realistic scenario, the blind flood protocol cannot be efficient anymore, since the radio channels (for instance in an urban environment) can be heavily loaded and the periodically initiated broadcasts can cause many collisions and even broadcast storms. In [11] communication is supposed to be limited, possibly subject to random failures, and asynchronous. Two different scenarios have been identified: a low and a high rate scenario. In the *high communication rate* scenario it is assumed, that the number of exchanged messages can be significantly larger than the updating command rates of the nodes, while in the *low communication rate* scenario this relation is the opposite. The command rate in this case is the frequency of the control algorithm, meaning that new control output is generated with the command rate. Nodes have control variables, used when calculating the control output for each node. These variables can be altered by the nodes and periodic updates are needed to accurately approximate their neighbours' states (position, velocity, etc.). Based on these assumptions, an adaptive algorithm was proposed: in the *high communication rate* phase it switches to a vanishing step-size gradient based distributed algorithm to approximate the appropriate control output, where the agent updates its control variables according to a local gradient descent. At the end of the updating phase some randomly selected agents transmit their control variables to their neighbours. In the *low communication rate* phase it is no longer possible to exactly determine the control variables (since messages cannot be transferred fast enough), therefore, a simulation based approximation is employed. Random gossip (namely Pairwise Gossip [8] and Broadcast Gossip [5]) models were used in the paper, since they allow to accurately model the randomness of the links between agents and do not rely on any possibly constraining scheduling of the communication between agents.

Table 5.1 summarises the solutions discussed in this section. The communication between the nodes can be passive and/or active. Passive means that there is no data transferred via wireless interface, thus the system depends only on data gathered from the sensors (such as infra-red sensors or ultra-sonic sensors). While in the active case the data is grouped into messages and disseminated over the wireless network formed by the nodes. In solutions where only passive communication is used no data is transferred and no transmission range defined. There are some cases however, where the passive system is used, together with a communication network. In those cases, the control algorithm needs additional data disseminated over the communication network. In the last column the transmission range is described for each solution.

In addition to the table it can be concluded that all the examined solutions (except Carlési's [11]) use an infinite communication medium. Therefore, there is no need for a medium access protocol since there is no packet losses and each node can transmit messages at any time regardless of the medium usage. This

assumption is essential to exchange information (such as position and velocity) with a very high frequency (*e.g.* 10 times in a second) and with the periodically received messages the control model can calculate different type of parameters from the received data (such as velocity form the position or acceleration from the velocity) with relatively high accuracy. Only the framework proposed by Carlési et al. [11] triestime slot to tackle the problems of communication failures, but still many idealistic assumptions are used for the communication environment, such as the radio range being infinite. The scalability of the network is usually not examined: when the flock gets bigger (consisting of more and more nodes), the number of the transmitted messages in a single timeslot increases as well, but since the communicational environment is almost perfect, the messages will not collide. The almost perfect communicational environment is assumed in some of the algorithms [11, 51], however an error is added to the received data, but it is utilised at control layer and not at the communication layer.

To summarise the section, it can be concluded that the current solutions cannot be used in real life use cases, since the communication environments being used do not meet the real life requirements. This is one of the main reasons why nearly all of these solutions work well only in simulations. For example, the algorithm proposed in [51] can achieve flocking only in a rural area, where the external radio noises are much lower than in an urban environment.

6. Closer to real life: simulation and demonstration of flocking systems. As it can be seen from the previous sections, there is an abundance of theoretical models and calculations to describe the properties of collective systems. But what happens when applying such models to real use cases? When examined, only a smaller fraction of the works from the literature introduce simulation measurements and experiments to study the stability and scalability under realistic environmental conditions (for example in the presence of unpredictable environmental noises), the majority of the solutions are not tested in real use cases, remaining only a mathematical basis for further research. The papers with real use cases conduct a systematic set of experiments using both physical and simulated robots, vehicles and UAVs, analysing the characteristics of the flocking solutions and checking if the results are in accordance with the ones that were predicted by the theoretical models. In this section we would like to provide an overview of the existing simulators and robots used to prove the applicability of the previously described algorithms.

Many of the simulators found in the literature are more or less applicable as a real-life simulator. In [34, 1], a dynamic model (the Webots simulator [30]) of the Khepera III robot was used. In the simulation the robots are equipped with distance sensors (infra-red and ultra-sonic), thus they can read the relative positions of their neighbours. The estimation is performed by a relative positioning system, based on the strength of the infrared signals interchanged by neighbouring robots. Robots are able to communicate via short messages, exchanging a unique id, necessary for the proposed algorithms of the system. They can also communicate periodically using the IEEE 802.11 wireless standard and UDP messages. Both communication systems make up two parallel networks, whose links are limited by the distance between robots in order to keep communications local. Messages are always sent from one robot to one of its communication neighbours. The framework allows robots to move in groups, harmonising the group velocity and avoiding obstacles by changing the direction of the group and the inter-robot distance. This is achieved by the robots by agreeing on a reference orientation without the use of any external reference, compass or global positioning system. As a result, robots share a distributed virtual compass, agreeing on a reference orientation. The framework is also equipped with a module enabling the group of robots to split, giving the decision on when to split and in which manner arbitrarily by one of the robots. The introduced framework can be implemented on any mobile robot capable of calculating the relative positions of adjacent robots and communicating with them. The simulator used for the simulated experiments was Webots, using a dynamic model of the Khepera III robot. The Webots simulator is a very realistic simulator, since the implementations can be transferred to real robots directly, but its disadvantage is that the robots can only move in a 2D plane because of the Khepera III specification. Up to 40 Khepera III robots were used in simulation, demonstrating experimentally the scalability with an increasing number of robots. The authors did not stop at the simulation level, they have further developed and tested a framework for the collective movement of mobile robots. The sets of experiments with real robots were as similar as possible to those in the simulation in order to compare them.They have performed experiments with real robots (using eight from the above mentioned Khepera III robots), proving that the framework works properly in a realistic environment. The join/split capabilities of the framework have been successfully tested both in simulation and

using real robots. The framework cannot deal with movement in three dimensions, however the authors claim that by modifying several modules it could be accomplished.

Another robot based simulator was presented in [13]. This physics-based simulator, called Controllable-Swarm Simulator (CoSS) [50], contains virtual Kobot based robots, which have two differentially driven motors and infrared (IR) sensors (the infrared short-range sensing system, which is composed of 8 IR sensors placed 45° intervals) around the base. Crosstalk among the nearby robots is avoided using the CSMA-CA (carrier sense multiple access collision avoidance) algorithm during sensing. For the wireless communication an IEEE 802.15.4/ZigBee module is used, with a range of around 20 m indoors. A virtual heading sensor (VHS) is utilised, composed of a digital compass and a wireless communication module, allowing the robots to sense the approximate relative orientations of nearby robots. The heading information is broadcasted by the robots to other robots within their communication range, converting the headings of the neighbouring robots to a local reference frame of the robot by vectorial subtraction of the robots own heading, for aligning the robot with its neighbours. The CoSS was used for performing experiments with more robots than physically available and for longer durations than possible in the physical experimental setups. It is using the ODE (Open Dynamics Engine) physics engine to study flocking behaviour with a large group of robots. Beside the simulated robots, they have carried out a wide range of experiments utilising physical robots, analysing the transient and steady-state characteristics of steered flocking, showing that the results conform with the theoretical models introduced earlier. The swarm of these mobile robots, connected via proximal sensing, were able to wander in an environment by moving as a coherent group in open space and avoiding obstacles.

The ARGoS simulator [38] is a bit different from the previously described solutions. Its main design focus is pointed at the simulation of large heterogeneous swarms of robots. Heterogeneity means in this case that there are different types of robots, which are responsible for special tasks e.g. a hand-bot [7] is used to climb a shelf and take something, while the eye-bots [41] are designed for coordination. The main advantage of this simulator is the open source implementation, together with the possibility to utilise various types of robots for the given use cases. The latter is achieved by dividing the simulated space into non-overlapping sub-spaces, each controlled by a separate physics engine. The rules implemented in each physics engine can be tailored to optimise the run-time of an experiment. Sensors and actuators are plug-ins that access the state of the simulated 3D space. Sensors are granted read-only access to the simulated 3D space, while actuators are allowed to modify it. ARGoS provides an abstract control interface that controllers must use to access sensors and actuators. The same control interface is also implemented on the real robots. In this way, the user code developed in simulation can be transferred to the real robots without modifications. The multi-threaded architecture of ARGoS can provide an astonishing performance of accurate 2D dynamics simulations of 10,000 robots in 60% of real time, furthermore accurate 3D dynamics simulations with the same number of robots in about real time. Because of its properties, it is widely used such as in [17, 6, 41, 7, 32].

A novel realistic simulation framework was introduced in [52] and used in [51]. It was developed to test and optimise two of their algorithms of collective motion: a self-propelled, bio-inspired flocking algorithm and a target tracking setup before employing them in a real scenario. The framework takes into account many of the inaccuracies of a real system, such as the position and velocity measurement error, furthermore the general Gaussian noise corresponding to the environmental effects, the low update rate of the GPS, the limited range of the communication, etc. The stability of the two algorithms was analysed thoroughly and it was discovered that the instabilities can be reduced with optimal strength of the viscous friction-like term. The optimised algorithms were tested on a group of real autonomous robots (quadcopters with on-board computer, GPS device and XBee communication module). They have presented a decentralised multi-copter flock that performs stable autonomous outdoor flight with up to 10 flying agents without central data processing or control.

For flocking algorithms to be applicable in real life use cases, experiments performed on real devices are essential. 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.

7. Conclusion. Despite of the large number of flocking algorithms already presented in the literature, only a few of them are capable for real world use cases, since there are many requirements that have to be satisfied by an algorithm to enable flocking in a real world environment. This was the reason that the main focus of our survey revolves around the applicability of the published flocking algorithms in real life. We have

tried to approach this complex question by categorising the available works based on how elaborate the used model is and trying to find some kind of evolution article by article.

First we have also presented a comprehensive overview of the control schemes involved in flocking systems, which are responsible for directing the flock to the desired destination and managing the shape of the group. Constructing an appropriate control mechanism to be feasible in real life is a huge and complex task, therefore, there are some widely used patterns which are the basis of almost every flocking algorithm.

We have also categorised the models introduced by various works, as the model constrains the control algorithm, influences the capabilities of the system and is one of the most important factors when designing a control algorithm. The models we have reviewed are based on the same assumption, that the vehicle or flock member is particle-like and it can propel itself in the direction and with the velocity desired based on the used control algorithm. All these models resemble the real world, dealing with increasing complexity in the model, external disturbances and nonlinear intrinsic dynamics. A common shortcoming of the examined solutions is that the vehicles in the flock or the formation may have additional constraints missing from the used models.

Furthermore the various communication methods were examined, and it can be concluded that the vast majority of the solutions assume an almost perfect communication environment, where there are no packet losses, the links are symmetric or the radio range is infinite. This is one of the main reasons why nearly all of these solutions work well only in simulations.

It can also be concluded that only a small fraction of the works from the literature introduce simulations and experiments to study the stability and scalability under realistic environmental conditions. The majority of the solutions are not tested in real use cases, remaining only a mathematical basis for further research. The real use cases were also introduced, providing an overview of the existing simulators and robots which are used to prove the applicability of the theoretical algorithms.

REFERENCES

[1] I. Navarro, *Exploring the split and join capabilities of a robotic collective movement framework*, Int J Adv Robotic Sy, 10 (2013).

[2] A. Abdessameud and A. Tayebi, *On consensus algorithms for double-integrator dynamics without velocity measurements and with input constraints*, Systems & Control Letters, 59 (2010), pp. 812–821.

[3] ———, *On consensus algorithms design for double integrator dynamics*, Automatica, 49 (2013), pp. 253–260.

[4] G. Antonelli, F. Arrichiello, and S. Chiaverini, *Flocking for multi-robot systems via the null-space-based behavioral control*, Swarm Intelligence, 4 (2010), pp. 37–56.

[5] T. C. Aysal, M. E. Yildiz, A. D. Sarwate, and A. Scaglione, *Broadcast gossip algorithms for consensus*, Signal Processing, IEEE Transactions on, 57 (2009), pp. 2748–2761.

[6] M. Bonani, V. Longchamp, S. Magnenat, P. Rtornaz, D. Burnier, G. Roulet, F. Vaussard, H. Bleuler, and F. Mondada, *The MarXbot, a Miniature Mobile Robot Opening new Perspectives for the Collective-robotic Research*, in International Conference on Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ, IEEE International Conference on Intelligent Robots and Systems, IEEE Press, 2010, pp. 4187–4193.

[7] M. Bonani, S. Magnenat, P. Rétornaz, and F. Mondada, *The hand-bot, a robot design for simultaneous climbing and manipulation*, in Intelligent Robotics and Applications, Springer, 2009, pp. 11–22.

[8] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, *Randomized gossip algorithms*, Information Theory, IEEE Transactions on, 52 (2006), pp. 2508–2530.

[9] M. Camperi, A. Cavagna, I. Giardina, G. Parisi, and E. Silvestri, *Spatially balanced topological interaction grants optimal cohesion in flocking models*, Interface focus, 2 (2012), pp. 715–725.

[10] Y. Cao and W. Ren, *Distributed coordinated tracking with reduced interaction via a variable structure approach*, Automatic Control, IEEE Transactions on, 51 (2012), pp. 33–48.

[11] N. Carlési and P. Bianchi, *Distributed coordination of a formation of heterogeneous agents with individual regrets and asynchronous communications*, in Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on, IEEE, 2012, pp. 3504–3511.

[12] A. Cavagna, L. Del Castello, I. Giardina, T. Grigera, A. Jelic, S. Melillo, T. Mora, L. Parisi, E. Silvestri, M. Viale, et al., *Flocking and turning: a new model for self-organized collective motion*, arXiv preprint arXiv:1403.1202, (2014).

[13] H. Çelikkanat and E. Şahin, *Steering self-organized robot flocks through externally guided individuals*, Neural Computing and Applications, 19 (2010), pp. 849–865.

[14] L. Cheng, Z.-G. Hou, M. Tan, Y. Lin, and W. Zhang, *Neural-network-based adaptive leader-following control for multiagent systems with uncertainties*, Neural Networks, IEEE Transactions on, 21 (2010), pp. 1351–1358.

[15] J. S. Coleman, *Foundations of social theory*, Harvard University Press, 1994.

[16] M.-C. Fan, Z. Chen, and H.-T. Zhang, *Velocity-free compact rigid flock control with input saturations*, in Control and Decision Conference (CCDC), 2013 25th Chinese, IEEE, 2013, pp. 316–321.
[17] E. Ferrante, W. Sun, A. Turgut, M. Dorigo, M. Birattari, and T. Wenseleers, *Self-organized flocking with conflicting goal directions*, in Proceedings of the European Conference on Complex Systems 2012, Springer, 2013, pp. 607–613.
[18] Z.-G. Hou, L. Cheng, and M. Tan, *Decentralized robust adaptive control for the multiagent system consensus problem using neural networks*, Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 39 (2009), pp. 636–647.
[19] S. Housheng and C. Guanrong, *Adaptive flocking with a virtual leader of multiple agents governed by nonlinear dynamics*, Control Conference (CCC), 2010 29th Chinese, (2010), pp. 5827–5832.
[20] A. Jadbabaie, J. Lin, and A. S. Morse, *Coordination of groups of mobile autonomous agents using nearest neighbor rules*, Automatic Control, IEEE Transactions on, 48 (2003), pp. 988–1001.
[21] Z. Jin, W. Xiaoqun, Y. Wenwu, S. Michael, and L. Jun-an, *Flocking of multi-agent dynamical systems based on pseudo-leader mechanism*, Systems & Control Letters, 61 (2012), pp. 195–202.
[22] P. Ke, S. Hou-Sheng, and Y. Yu-Pu, *Coordinated control of multi-agent systems with a varying-velocity leader and input saturation*, Communications in Theoretical Physics, 52 (2009), p. 449.
[23] S.-J. Kim and I.-H. Whang, *Acceleration constraints for maneuvering formation flight trajectories*, Aerospace and Electronic Systems, IEEE Transactions on, 48 (2012), pp. 1052–1060.
[24] A. Kumar, S. Sharma, R. Tiwari, and S. Majumdar, *Area exploration by flocking of multi robot*, Procedia Engineering, 41 (2012), pp. 377–382.
[25] N. E. Leonard and E. Fiorelli, *Virtual leaders, artificial potentials and coordinated control of groups*, in Decision and Control, 2001. Proceedings of the 40th IEEE Conference on, vol. 3, IEEE, 2001, pp. 2968–2973.
[26] X. LI and J. CAO, *Adaptive synchronization for delayed neural networks with stochastic perturbation*, Journal of the Franklin Institute, 345 (2008), pp. 779–791.
[27] H. Lim and C. Kim, *Flooding in wireless ad hoc networks*, Computer Communications, 24 (2001), pp. 353–363.
[28] B. Liu and H. Yu, *Flocking in multi-agent systems with a bounded control input*, in Chaos-Fractals Theories and Applications, 2009. IWCFTA '09. International Workshop on, Nov 2009, pp. 130–134.
[29] I. S. MAGNETIZATION, *The spontaneous magnetization of a two-dimensional ising model*, Physical Review, 85 (1952).
[30] O. Michel, *Webotstm: Professional mobile robot simulation*, arXiv preprint cs/0412052, (2004).
[31] A. Mogilner and L. Edelstein-Keshet, *A non-local model for a swarm*, Journal of Mathematical Biology, 38 (1999), pp. 534–570.
[32] F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J.-C. Zufferey, D. Floreano, and A. Martinoli, *The e-puck, a robot designed for education in engineering*, in Proceedings of the 9th conference on autonomous robot systems and competitions, vol. 1, IPCB: Instituto Politécnico de Castelo Branco, 2009, pp. 59–65.
[33] D. J. Naffin and G. S. Sukhatme, *Negotiated formations*, in Proceedings of the eighth conference on intelligent autonomous systems, 2004, pp. 181–190.
[34] I. Navarro and F. Matía, *A framework for the collective movement of mobile robots based on distributed decisions*, Robotics and Autonomous Systems, 59 (2011), pp. 685–697.
[35] A. Okubo, *Dynamical aspects of animal grouping: swarms, schools, flocks, and herds*, Advances in biophysics, 22 (1986), pp. 1–94.
[36] R. Olfati-Saber, *Flocking for multi-agent dynamic systems: Algorithms and theory*, Automatic Control, IEEE Transactions on, 51 (2006), pp. 401–420.
[37] J. K. Parrish, S. V. Viscido, and D. Grünbaum, *Self-organized fish schools: an examination of emergent properties*, The biological bulletin, 202 (2002), pp. 296–305.
[38] C. Pinciroli, V. Trianni, R. OGrady, G. Pini, A. Brutschy, M. Brambilla, N. Mathews, E. Ferrante, G. Di Caro, F. Ducatelle, et al., *Argos: a modular, parallel, multi-engine simulator for multi-robot systems*, Swarm intelligence, 6 (2012), pp. 271–295.
[39] S. A. Quintero, G. E. Collins, and J. P. Hespanha, *Flocking with fixed-wing uavs for distributed sensing: A stochastic optimal control approach*, in American Control Conference (ACC), 2013, IEEE, 2013, pp. 2025–2031.
[40] C. W. Reynolds, *Flocks, herds and schools: A distributed behavioral model*, in ACM SIGGRAPH Computer Graphics, vol. 21, ACM, 1987, pp. 25–34.
[41] J. Roberts, T. Stirling, J.-C. Zufferey, and D. Floreano, *Quadrotor using minimal sensing for autonomous indoor flight*, in Proceedings of the European Micro Air Vehicle Conference and Flight Competition (EMAV2007), 2007.
[42] H. Shi, L. Wang, and T. Chu, *Coordinated control of multiple interactive dynamical agents with asymmetric coupling pattern and switching topology*, in Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on, IEEE, 2006, pp. 3209–3214.
[43] ———, *Virtual leader approach to coordinated control of multiple mobile agents with asymmetric interactions*, Physica D: Nonlinear Phenomena, 213 (2006), pp. 51–65.
[44] H. Shi, L. Wang, and T. Chu, *Virtual leader approach to coordinated control of multiple mobile agents with asymmetric interactions*, Physica D, 213 (2006), pp. 51–65.
[45] W. M. Spears, D. F. Spears, J. C. Hamann, and R. Heil, *Distributed, physics-based control of swarms of vehicles*, Autonomous Robots, 17 (2004), pp. 137–162.
[46] H. Su, G. Chen, X. Wang, and Z. Lin, *Adaptive second-order consensus of networked mobile agents with nonlinear dynamics*, Automatica, 47 (20111), pp. 368–375.
[47] H. Su, X. Wang, and Z. Lin, *Flocking of multi-agents with a virtual leader, part i: with a minority of informed agents*, Proc. the 46th IEEE Conference on Decision and Control, (December 2007), pp. 2937–2942.
[48] ———, *Flocking of multi-agents with a virtual leader, part ii: with a virtual leader of varying velocity*, Proc. the 46th IEEE

Conference on Decision and Control, (December 2007), pp. 1429–1434.
[49] H. Su, N. Zhang, M. Z. Chen, H. Wang, and X. Wang, *Adaptive flocking with a virtual leader of multiple agents governed by locally lipschitz nonlinearity*, Nonlinear Analysis: Real World Applications, 14 (2013), pp. 798 – 806.
[50] A. E. Turgut, H. Çelikkanat, F. Gökçe, and E. Şahin, *Self-organized flocking in mobile robot swarms*, Swarm Intelligence, 2 (2008), pp. 97–120.
[51] G. Vásárhelyi, C. Virágh, G. Somorjai, N. Tarcai, T. Szörényi, T. Nepusz, and T. Vicsek, *Outdoor flocking and formation flight with autonomous aerial robots*, arXiv preprint arXiv:1402.3588, (2014).
[52] C. Virágh, G. Vásárhelyi, N. Tarcai, T. Szörényi, G. Somorjai, T. Nepusz, and T. Vicsek, *Flocking algorithm for autonomous flying robots*, Bioinspiration & biomimetics, 9 (2014), p. 025012.
[53] M. Wang, H. Su, M. Zhao, M. Z. Chen, and H. Wang, *Flocking of multiple autonomous agents with preserved network connectivity and heterogeneous nonlinear dynamics*, Neurocomputing, (2013), pp. 169–177.
[54] Q. Wang, H. Fang, J. Chen, Y. Mao, and L. Dou, *Flocking with obstacle avoidance and connectivity maintenance in multi-agent systems*, in Decision and Control (CDC), 2012 IEEE 51st Annual Conference on, IEEE, 2012, pp. 4009–4014.
[55] X. F. WANG, *Complex networks: Topology, dynamics and synchronization*, International Journal of Bifurcation and Chaos, 12 (2002), pp. 885–916.
[56] G. Wen, Z. Duan, H. Su, G. Chen, and W. Yu, *A connectivity-preserving flocking algorithm for multi-agent dynamical systems with bounded potential function*, IET Control Theory & Applications, 6 (2012), pp. 813–821.
[57] W. Yu, J. Cao, and J. Lü, *Global synchronization of linearly hybrid coupled networks with time-varying delay*, SIAM Journal on Applied Dynamical Systems, 7 (2008), pp. 108–133.
[58] W. Yu, G. Chen, and M. Cao, *Distributed leaderfollower flocking control for multi-agent dynamical systems with time-varying velocities*, Systems & Control Letters, 59 (2010), pp. 543–552.
[59] W. Yu, G. Chen, M. Cao, and J. Kurths, *Second-order consensus for multiagent systems with directed topologies and nonlinear dynamics*, IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society, 40 (2010), pp. 881–91.
[60] W. Yu, G. Chen, and J. Lü, *On pinning synchronization of complex dynamical networks*, Automatica, 45 (2009), pp. 429–435.
[61] W. Yu, W. X. Zheng, G. Chen, W. Ren, and J. Cao, *Second-order consensus in multi-agent dynamical systems with sampled position data*, Automatica, 47 (2011), pp. 1496–1503.
[62] H. Zhang, J. Llorca, C. C. Davis, and S. D. Milner, *Nature-inspired self-organization, control, and optimization in heterogeneous wireless networks*, Mobile Computing, IEEE Transactions on, 11 (2012), pp. 1207–1222.
[63] H.-T. Zhang, C. Zhai, and Z. Chen, *A general alignment repulsion algorithm for flocking of multi-agent systems*, Automatic Control, IEEE Transactions on, 56 (2011), pp. 430–435.
[64] Y. Zheng and L. Wang, *Consensus of heterogeneous multi-agent systems without velocity measurements*, International Journal of Control, 85 (2012), pp. 906–914.
[65] J. Zhou, X. Wu, W. Yu, M. Small, and J.-a. Lu, *Flocking of multi-agent dynamical systems based on pseudo-leader mechanism*, Systems & Control Letters, 61 (2012), pp. 195–202.
[66] A.-M. Zou, K. D. Kumar, and Z.-G. Hou, *Attitude coordination control for a group of spacecraft without velocity measurements*, Control Systems Technology, IEEE Transactions on, 20 (2012), pp. 1160–1174.

Edited by: Giacomo Cabri and Emma Hart
Received: Dec 15, 2014
Accepted: Jul 10, 2015