A Minimalist Flocking Algorithm for Swarm Robots

Christoph Moeslinger[1], Thomas Schmickl[1], and Karl Crailsheim[1]

Karl-Franzens University Graz, Artificial Life Lab of the Department of Zoology, Universitätsplatz 2, 8010 Graz, Austria

{christoph.moeslinger, thomas.schmickl}@uni-graz.at

Abstract. In this paper we describe a low-end and easy to implement flocking algorithm which was developed for very simple swarm robots and which works without communication, memory or global information. By adapting traditional flocking algorithms and eliminating the need for communication, we created an algorithm with emergent flocking properties. We analyse its potential of aggregating an initially scattered robot swarm, which is not a trivial task for robots that only have local information.

1 Introduction

One of the most amazing developments in biological evolution is the domain of social insects. These animals, although being very small, achieve impressive feats. Bees, ants and termites live in elaborately constructed nests which are, in comparison to the insect, gigantic [1]. The colony super-organism can be characterized as being *swarm-intelligent* [2] because its abilities, for example to optimally allocate foragers to food sources, are a result of the interactions within the swarm and cannot be achieved by the single individual. This decentralized and distributed way of achieving a goal is an interesting and useful field of study which has inspired the fields of swarm-intelligence [3, 4] and swarm robotics [5]. In the last decade, a lot of control strategies and algorithms for robotic swarms have been presented, both in simulated and real robot swarms.

Innovation and industrial progress make it possible to manufacture such swarm robots in smaller and smaller sizes [6]. For such small robots, reaching a common goal is not an easy task. This is due to the constraints that come with decreasing size, like small sensor ranges, very limited computational power, little memory and imprecise locomotion. Thus, developing control algorithms for small scale swarm robots raises interesting questions, like "how can a robot, which only knows about the very small part of the environment around itself, achieve a common goal with the rest of the swarm?" and "given all constraints, what is the most efficient way to reach a common goal?".

Reaching such a common goal often implies that the swarm has to be aggregated for cooperative work, transport [7, 8] or assembly [9, 10]. This aggregation task seems to be relatively trivial, but the constraints of a swarm of small robots

create some conceptual problems. Small robots usually do not have the capability of long range communication and do not possess global information like position and heading. When only local information is available, the coordination of numerous autonomous agents requires a different approach. Examples of such an approach already exist in nature: the phenomenon of flocks, herds and schools.

Flocks can be aggregations of up to several thousand individuals which move together with astounding elegance and flexibility. Craig Reynolds was amongst the first to abstract this behaviour to steer a swarm of simulated birds which he called *boids* [11]. To do this, he implemented three behavioural rules in his autonomous boids: *collision avoidance* to evade obstacles and flock mates which are too close, *flock centering* to stay close to flock mates and *velocity / heading matching* to move in the same direction as nearby flock mates. The resulting simulated flocks appear very similar to real flocks. As a result, several approaches to adapt this behaviour to a robotic swarm have been made. Although these pursues were successful in creating flocks, all of them incorporated something that is usually not found in real bird flocks or fish schools: communication and knowledge of the own heading [12–15] or predefined leaders [16]. Other interesting approaches used light beacons (and the resulting shadows) in an arena to generate a flocking swarm, either using evolved neural networks [17] or adapted aggregation algorithms in combination with *situated communication* [18], where the only significance is the presence or absence of an "empty" message. Usually, the matching of velocity and heading was only possible when the robots were able to communicate with their flock mates through a stable communication channel. This means that each robot needed to know its own heading (through a digital compass) and the headings and speeds of all its (near) flock mates.

This involves extensive communication, which can be far too complex for small and numerous swarm robots, and, more importantly, that is also not in accordance to how a real flock achieves alignment. In real swarms, the animals identify the heading of each other because they derive it visually from their flock mates' body form, although most fish species also have a specialized *lateral line organ* which is additionally used for alignment [19]. Such methods could be adapted for swarm robots by using on-board cameras and image recognition, but that would probably be way too complex for very small robots.

We found out that there is also a much easier way to create flocking behaviour. By discretizing the robots' sensor fields into sectors and using different distance thresholds for attraction and repulsion in these sectors, robot swarms can achieve *emergent* alignment.

2 Material & Methods

2.1 Algorithm Requirements

The main aim of our flocking algorithm is simplicity. This means that we developed our algorithm in consideration of very limited computational power and

only minimalist swarm robot equipment. Such a basic equipment is a set of distance sensors, which are usually used for collision avoidance. We wanted to utilize just these sensors to generate complex swarm behaviours.

Swarm robots usually have IR-sensors in all directions which enable the detection of reflecting surfaces like walls, obstacles and other robots. A major disadvantage of these IR-sensors is, that they have very limited range and cannot discriminate robots from obstacles, except when using a combination of active and passive sensing. Here, active sensing means that the robot activates its IR-light at the position of the sensor and checks for reflections, whereas passive sensing means that the robot only checks for IR-light from other robots without emitting IR-light itself. Usually, the brighter the sensed IR-light, the higher is the value that the sensor returns. These IR-sensor values are all that is needed for the effectivity of our algorithm. Of course other distance measuring sensors, like ultrasonic sensors, can be used for our algorithm as well.

The minimal requirements of the algorithm are 4 circumferential distance sensors with limited range and 3 discrete reactions in movement: *move straight*, *turn left* or *turn right*. The algorithm does not require any global information about positions or headings, precise sensor information, memory, elaborate robot-to-robot recognition or communication.

2.2 Flocking Algorithm

Each robot in the swarm periodically emits IR-pulses. The robots then react (*move straight*, *turn left* or *turn right*) depending on information from their active and passive IR-sensors. These sensors are polled periodically and the returned values are then checked against predefined thresholds (Fig. 1B) in a simple subsumption architecture (Fig. 1A).

First, the active IR-value for the front sensor is polled to find out whether there is an obstacle in front. If the value for the reflected IR-light is above a certain threshold, the robot turns away in a random direction. This is the basic *collision avoidance* of our robots.

If there are no objects in its way, the robot checks the passive IR-values of all sensors. If the front, left or right sensor is above a certain threshold, the robot turns away from what is presumably another robot which is too close. This rule is usually referred to as the *separation rule* in flocking algorithms.

If there is no other robot too close, the robot checks the passive IR-values of its left, right and rear sensors. For every sensor that returns a value that is above the environmental IR-light threshold but below the threshold which defines the maximally desired distance to another robot in that sector, the robot performs a basic *vector addidtion* and adds up all turns. It then decides to turn in a direction depending on whether there were more left or more right turns. Robots in the rear zone trigger a random turn reaction. This rule is usually referred to as the *cohesion rule* in flocking algorithms.

The third rule in flocking algorithms is usually the *alignment rule* which generates the common direction of movement in a flock. Since we wanted our

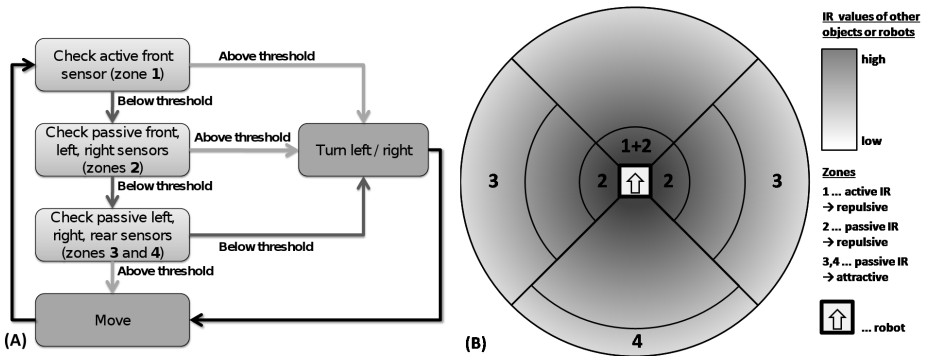

Fig. 1. A: Simple subsumption architecture depicting the flocking algorithm. The first decision results in *collision avoidance*, the second decision results in *robot separation* and the third decision results in *flock cohesion* and *emergent alignment*. B: Simplified depiction of the perceived IR-values of other objects (reflected active IR) or flock mates (passive IR) dependent on their distance to the robot. Thresholds and the resulting zones for a robot with 4 IR-sensors.

algorithm to be as simple as possible we wanted to exclude complex communication or image recognition procedures and implemented a method which generates *emergent* alignment. To achieve this we adjusted the thresholds for the *cohesion rule* so that robots tend to follow other robots. This is done by simply shifting the threshold for the rear sensor more outwards in comparison to the thresholds for the left and right sensors (zones 3 and 4 in Fig. 1B). Depending on the position and heading of two approaching robots, one robot will be behind the other robot. When both robots move, the robot behind will turn towards the robot in front *before* the robot in front reacts and turns around. This creates a *leader robot* and a *follower robot*, purely by chance. These two robots will then move around in the arena without separating. If the path of these two robots is blocked by an obstacle or another robot joins the flock, the arrangement can change instantly. If two robots approach frontally, they will avoid each other, only to turn back to each other shortly after, which can create a *deadlock* situation. To prevent such situations, we implemented a random-turn reaction which means that robots will randomly turn either left or right when avoiding other robots in front (zone 1 in Fig. 1B).

2.3 Simulator

We conducted simulations firstly as a *proof-of-concept* and secondly to evaluate the aggregation and flocking capability of a robot swarm which uses our algorithm. These simulations use very abstracted and ideal two-dimensional models of autonomous mobile agents with circumferential sensors. For our simulations we used a custom-built simulator (Fig. 2A) based on the multi-agent programming language NetLogo [20]. A short video can be seen at [21].

The tests were conducted with different numbers of robots in differently sized arenas. For generality, we assumed 1 robot-diameter as the unit of measurement. A simulated robot moved with 3 units per second and polled its distance-sensors, which had a maxmium range of 5 units, 60 times per second. The thresholds were set to 1 unit for zone 1, 1 unit for zones 2, 2 units for zones 3 and 3.5 units for zone 4. These values were derived from tests with real swarm robots [22]. To emulate real-world conditions and to avoid certain deadlocks (e.g., two robots circling each other) we introduced inaccuracies. In the simulations, the sensor values were subject to a 5% random-normal error in measurement and the speed of the individual robots was subject to a 5% random-normal variance. We simulated swarms of 5, 10, 15, 20 and 25 robots in bordered square arenas with the sizes of 3600, 7200, 10800, 14400 and 18000 robot-diameter^2. The sizes of the arenas were chosen to create density-neutral setups so that we could measure the impact of swarm size on the efficiency of the algorithm.

3 Results

Our simulations contribute to two distinct analyses. On the one hand, we wanted to investigate the flocking capabilities of our swarms and, on the other hand, we wanted to determine the algorithm's efficiency to let randomly scattered robots form an aggregation in an arena. Here, *aggregated* or *flock* means that we counted each robot which was inside the IR-sensor field of another robot and counted the total number of robots within that *connected* swarm.

3.1 Flocking Analysis

Our first simulations were set up to find out the mobility of an aggregated robot swarm. For this we simulated a huge 300x300 robot-diameter^2 arena so that the robots were not confined in their movement by the borders. An already aggregated swarm of 1 to 25 robots with randomized headings was placed into the middle of this arena. We then added up the movements of all robots for 60 seconds and calculated the average distance covered by a robot in the swarm. Additionally, we measured the movement of the coherent flock by calculating its center of mass and adding up its path.

An ideally mobile flock would cover the same distance as the average distance covered by a robot in the flock. Since our algorithm does merely impede but not prevent a flock from splitting up, we only counted simulation runs where the whole swarm stayed together the whole time of the simulation, which was the case, on average, in 46% of all runs. Each experiment was repeated until there were 10 successful runs. A comparison of the distance measurements for the robots and the center of mass of the flock can be seen in Fig. 2B. The average movement of a robot in a flock differs slightly due to the implemented speed error.

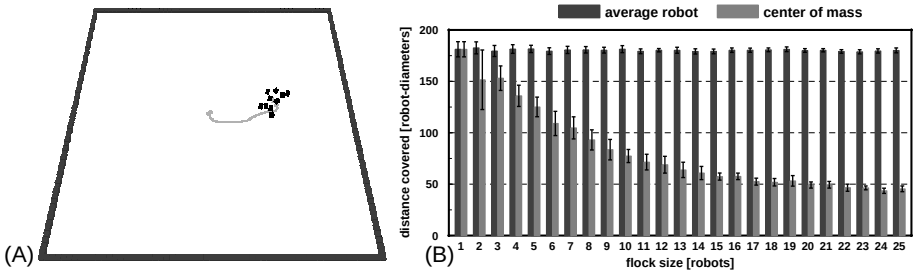

Fig. 2. A: Screenshot of our simulator with a swarm of 10 robots (black cubes) in a small arena with borders. The path of the center of mass of the flock is depicted by a grey line. B: Average movement of a robot in a flock (dark grey) in comparison to the average movement of the center of mass of the flock (light grey) in 60 seconds. Means and standard deviations of 10 repetitions.

3.2 Aggregation Analyses

General Efficiency: Next, we focussed on finding out the aggregation efficiency of a robot swarm which uses our algorithm. In these simulations the robots had been randomly scattered in an arena and had to aggregate. There was no global information about where to aggregate and the robots reacted purely on short-range IR-detection. A swarm was considered as being aggregated when at least 60% of its members were in the same flock. We measured the time it took a scattered swarm to aggregate under different combinations of swarm size and arena size. 100 repetitions with randomized starting positions and headings were evaluated, the result can be seen in Fig. 3A.

Density-neutral Efficiency: Further simulations were conducted only under density-neutral conditions, for example 5 robots in a 3600 robot-diameter^2 arena or 10 robots in a 7200 robot-diameter^2 arena. This was done to measure the influence of swarm size on the efficiency of the algorithm. A supplementary measurement was made by increasing the desired aggregation size from 60% to 80% (Fig. 3B). Each simulation was evaluated by the median, first and third quartiles of 100 repetitions.

4 Discussion

The flocking analysis showed that the mobility of a flock of swarm robots which use our algorithm was dependent on the flock's size. A small flock of 5 robots still moved 69% of the distance a single robot could have moved, whereas increasing the flock size to up to 25 robots decreased the mobility down to 25%. This decrease is of course a major drawback in comparison to traditional flocking algorithm implementations, where large flocks can still be quite mobile.

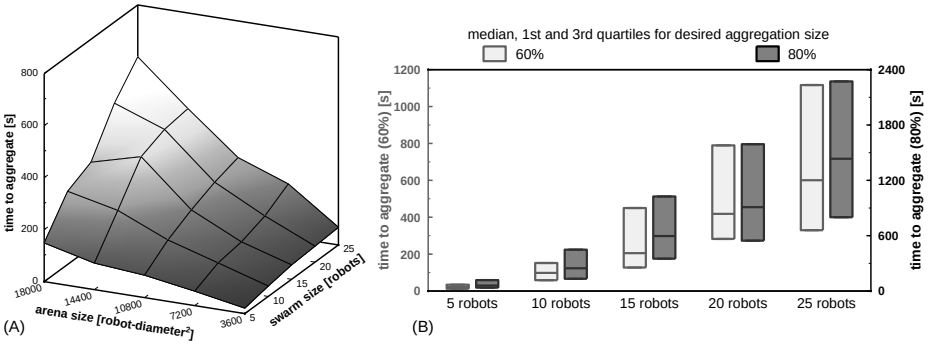

Fig. 3. A: Aggregation speed analysis for all swarm sizes and all arena sizes. We measured the median time it took for an initially scattered robot swarm to form an aggregation of at least 60% of the whole swarm. B: Comparison of aggregation time for an initially scattered robot swarm in density neutral setups for 60% (light grey) and 80% (dark grey) desired aggregation size. Median, first and third quartiles of 100 repetitions.

The aggregation analyses suggested that our flocking algorithm was quite efficient concerning aggregating scattered robots. It should be noted that, contrary to our expectations, there was an increase in aggregation time when increasing the number of robots in the same arena. In retrospective this can be explained by the flocking analysis, which showed that with increasing numbers of flock mates, the movement of the flock was more and more limited. Thus it became harder for two larger flocks to merge in order to attain the desired aggregation size.

Our density-neutral experiments showed that the aggregation time was increased when increasing the number of robots and the size of the arena. This is due to the larger distances that flocks have to cover in order to join other flocks and the aforementioned decrease of flock movement in bigger flocks. When the desired aggregation size was changed from 60% of the whole swarm to 80% of the whole swarm, there was a large increase in aggregation time. The reason why we did not test the goal of achieving 100% aggregation was that, when working with numerous swarm robots, one has to expect a small fraction of malfunctioning or stuck robots, so that *perfect* achievements are rather improbable.

Summing up, our flocking algorithm enables swarm robots to form a coherent and reactive flock which moves around in the arena randomly. Our results suggest that it works well with small swarms and is especially suited for robots with minimal equipment. It could, for instance, be used for the aggregation part of a more comprehensive scenario. Moreover, the algorithm can also be used for a heterogeneous robotic swarm when the robot types use the same distance-sensing method. Our next step will be to port the algorithm to a real swarm of up to 30 heterogeneous swarm robots [22, 23].

[1] Supported by: EU-IST-FET project 'SYMBRION', no. 216342; EU-ICT project 'REPLICATOR', no. 216240; EU-IST FET project 'I- Swarm', no. 507006.

References

1. Wilson, E.O.: The Insect Societies (Harvard Paperbacks). Belknap Press (November 1974)
2. Beni, G., Wang, J.: Swarm intelligence. In: Proc. of the Seventh Annual Meeting of the Robotics Society of Japan. (1989) 425–428
3. Kennedy, J., Eberhart, R.C.: Swarm Intelligence. Morgan Kaufmann (2001)
4. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm Intelligence: From Natural to Artificial Systems. Oxford Univ. Press (1999)
5. Dudek, G., Jenkin, M., Milios, E., Wilkes, D.: A taxonomy for swarm robots. Intelligent Robots and Systems '93 **1** (1993) 315–325
6. Seyfried, J., et al.: The I-SWARM project: Intelligent small world autonomous robots for micro-manipulation. In Şahin, E., Spears, W.M., eds.: Swarm Robotics - SAB 2004 Int. Workshop, Berlin, Germany, Springer-Verlag (2005) 70–83
7. Trianni, V., Groß, R., Labella, T., Şahin, E., Dorigo, M.: Evolving aggregation behaviors in a swarm of robots. Lect. Notes in Artifcial Intelligence **2801** (2003) 865–874
8. Martinoli, A., Easton, K., Agassounon, W.: Modeling swarm robotic systems: A case study in collaborative distributed manipulation. Int. Journal of Robotics Research **23**(4) (2004) 415–436
9. SYMBRION: Project website (2009) http://www.symbrion.eu/.
10. REPLICATOR: Project website (2009) http://www.replicators.eu/.
11. Reynolds, C.W.: Flocks, herds, and schools. Computer Graphics **21**(4) (1987) 25–34
12. Mataric, M.J.: Designing emergent behaviors: from local interactions to collective intelligence. Proc. of the Second Int. Conf. on From Animals to Animats 2: simulation of adaptive behavior (1993) 432–441
13. Turgut, A., Çelikkanat, H., Gökçe, F., Şahin, E.: Self-organized flocking in mobile robot swarms. Swarm Intelligence **2**(2) (December 2008) 97–120
14. Hayes, A.T., Dormiani-Tabatabaei, P.: Self-organized flocking with agent failure: Off-line optimization and demonstration with real robots. In: Int. Conf. on Robotics and Automation. (2002) 3900–3905
15. Balch, T., Hybinette, M.: Social potentials for scalable multi-robot formations. Volume 1. (2000) 73–80 vol.1
16. Kelly, I.D., Keating, D.A.: Flocking by the fusion of sonar and active infrared sensors on physical autonomous mobile robots. Proc. of the The Third Int. Conf. on Mechatronics and Machine Vision in Practice **1** (1996) 1–4
17. Baldassarre, G., Nolfi, S., Parisi, D.: Evolving mobile robots able to display collective behaviors. Artificial Life **9**(3) (2003) 255–267
18. Bjerknes, J.D., Winfield, A., Melhuish, C.: An analysis of emergent taxis in a wireless connected swarm of mobile robots. In: IEEE Swarm Intelligence Symposium, Los Alamitos, CA, IEEE Press (2007) 45–52
19. Partridge, B.L., Pitcher, T.J.: The sensory basis of fish schools: relative roles of lateral line and vision. Journal of Comparative Physiology **135**(4) (1980) 315–325
20. Wilensky, U.: Netlogo. Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL (1999)
21. Moeslinger, C.: Video link (2009) http://zool33.uni-graz.at/artlife/flocking.
22. Swarmrobot: Project website (2009) http://www.swarmrobot.org/tiki-index.php.
23. ePuck: e-puck desktop mobile robot - website (2009) http://www.e-puck.org/.