Erschienen in: ALIFE 2019 : The 2019 Conference on Artificial Life. - Cambridge, Massachusetts : MIT Press, 2019. - S. 642-649
https://dx.doi.org/10.1162/isal_a_00233

Collective Change Detection: Adaptivity to Dynamic Swarm Densities and Light Conditions in Robot Swarms

Mostafa Wahby[1], Julian Petzold[1], Catriona Eschke[1], Thomas Schmickl[2], and Heiko Hamann[1]
[1]Institute of Computer Engineering, University of Lübeck, Lübeck, Germany
[2]Artificial Life Lab, University of Graz, Graz, Austria
hamann@iti.uni-luebeck.de

Abstract

Robot swarms are known to be robust to individual robot failures. However, a reduced swarm size causes a reduced swarm density. A too low swarm density may then decrease swarm performance, that should be compensated by adapting the individual behavior. Similarly, swarm behaviors can also be adapted to changes in the environment, such as dynamic light conditions. We study aggregation of swarm robots controlled by an extended variant of the BEECLUST algorithm. The robots are asked to aggregate at the brightest spot in their environment. Our approach efficiently adapts this swarm aggregation behavior to variability in swarm density and light conditions. First, each robot individually monitors its environment continuously by sampling its local swarm density and perceived light condition. Second, we exploit the collaboration of robots by letting them share features of these measurements with their neighbors by communication. In extensive robot swarm experiments with ten robots we validate our approach with dynamically changing swarm densities and under dynamic light conditions. We find an improved performance compared to robot swarms without communication and without awareness of the swarm density.

Introduction

Swarm density has usually a significant impact on swarm performance in general and consequently on scalability (Hamann, 2018), for example, as observed in collective decision-making processes (Khaluf et al., 2017). Known advantages of swarm robotics, such as robustness and scalability, depend on a swarm's density and most swarm behaviors are sensitive to changes in swarm density. With large swarm sizes, more robots break, get lost, or may intentionally be removed. As an effect, the swarm density can decrease at runtime. Similarly, an operator may add robots to the system or the operating area may decrease over time, both effectively increasing the swarm density. Online changes in density have an impact on the swarm performance and adaptivity to these changes may prove to increase efficiency (Kernbach et al., 2013). Similarly, adapting the swarm behavior to dynamic environmental changes may be essential even for survival (Mallon et al., 2001) or may at least increase performance (Bonabeau et al., 1999).

In this paper, we make two main contributions that address two main challenges of designing swarm algorithms that adapt to dynamic densities and dynamic environments: (1) swarm density and environment are global features that can only be efficiently detected collectively; (2) for both, we face a tradeoff between filtering noise while reacting quickly to changes. In statistics, this problem is known as change detection (Picard, 1985) requiring to minimize the detection delay but also the cost of false alarms. Only here we require a decentralized implementation. Once a change in the swarm density is detected collectively, robots may be able to react individually. In the case of a too high density, robots may take themselves out of the game by leaving the operating area (Mayya et al., 2018). In the case of too low density, we get the more interesting challenge because we have to adapt the individual behavior appropriately, as studied here.

Previously, we have investigated the automatic adaptation of a robot swarm's behavioral features to the light settings in static environments (Wahby et al., 2016). This was based on an initial calibration phase by individual robots without exploiting collaboration between robots. However, certain environmental features may only be detected collectively (Schmickl et al., 2007) and it may be required to share information (Valentini et al., 2016). Here, we present an extension that adapts the robot behaviors to dynamic light conditions and exploits collaboration using communication.

Our approach is based on a site-specific aggregation behavior (Trianni et al., 2003; Soysal and Şahin, 2007). The robot swarm is supposed to aggregate at a spot of specific environmental conditions. Here, the aggregation spot is supposed to be the brightest spot. A robot can explore the whole arena, but it cannot know whether a yet unexplored brighter area exists or a previously explored area has increased its brightness meanwhile. Hence, a swarm robot should always stay explorative and as a sum of individual behaviors also the robot swarm should stay adaptive to changes.

As control algorithm we use BEECLUST (Schmickl and Hamann, 2011; Schmickl et al., 2008; Kernbach et al., 2009), a standard approach to aggregation in swarm robotics. BEECLUST is inspired by the behavior of young

honeybees (Szopek et al., 2013) that collectively find and select the warmest spot in a temperature gradient. According to the behavioral model they do that in a decentralized way and without any explicit communication. This behaviour was modeled and studied often in swarm robotics (Schmickl et al., 2009; Valentini and Hamann, 2015; Correll and Hamann, 2015), and a fuzzy-logic-based extension was proposed (Arvin et al., 2012, 2014), but with the original environmental feature of temperature being replaced by luminance. Here, the desired aggregation area is indicated by a light spot. We want a majority of the swarm to aggregate at the brightest light spot while a minority still stays adaptive as it keeps exploring the environment. We test a dynamic environment with changing light conditions (Schmickl et al., 2009). Initially, there are two peaks in the light distribution that we can see as local and global optima. It is essential that the swarm has a global awareness of the light distribution. A greedy search is not an option as the swarm needs to avoid both local optima and splitting between bright areas. The light conditions change three times during an experiment requiring three switches of the swarm's aggregation spot. The swarm robots need to explore the arena continuously and re-allocate quickly.

We implemented our proposed extension to the BEECLUST algorithm on Thymio II robots (Riedo et al., 2013) and tested our approach in 25 robot experiments of 24+ minutes. We analyze the obtained results and find that the adaptation to dynamic environments is effective, that sharing gathered information between robots by communication is advantageous, and that our adaptivity approach to dynamic swarm densities is also effective.

A Short History of BEECLUST

BEECLUST is a bio-inspired aggregation algorithm, derived from the navigation behavior of young honeybees in temperature gradients within beehives (Schmickl and Hamann, 2011; Schmickl et al., 2008; Kernbach et al., 2009; Hamann, 2018). The original behavioral model of bees is translated to a swarm robot controller where a light gradient is considered instead of a temperature gradient. Experimenting with light simplifies sensing compared to temperature measurements (temperature sensors are less sensitive, are fragile, and have longer latencies). The original BEECLUST algorithm is defined as follows:

1. A robot moves forward.
2. If a robot approaches the arena borders, it turns away to a random direction and continues with step 1.
3. If a robot meets another robot, it stops and measures the local luminance. The higher the luminance, the longer its waiting time is (i.e., the amount of time the robot stays stationary).
4. After the waiting time is over, the robot turns away to a random direction and continues with step 1.

The BEECLUST algorithm produces a characteristic swarm-level behavior: Initially several small robot clusters form across the environment. When the waiting time has elapsed, robots leave their cluster again and, hence, clusters may disappear. Clusters in brighter areas persist longer and have a higher probability to increase in size by being approached by additional robots. There is a positive feedback loop on cluster growth because bigger clusters have a bigger probability to grow. In addition, robots within a big cluster may not be able to leave after their waiting time has elapsed because they are physically blocked. This contributes as additional positive feedback. Finally, one or a few big clusters form. A few robots still leave the cluster, explore the possibly dynamic environment, and may come back, joining the cluster again.

The key point for the BEECLUST algorithm to succeed is the correct mapping of environmental conditions to the resulting waiting times. Proper waiting times depend on environmental features, for example, the light distribution, and the swarm density (i.e., the number of robots in a given area). A lower swarm density means that robots are less likely to approach a robot cluster, hence, longer waiting times would compensate for this effect and ensure that clusters still have a good chance of growing. High swarm densities may result in a limited competition between clusters, such that clusters at suboptimal positions do not dissolve as quickly as desired. Similarly, bright robot arenas may in average give too long waiting times and dark arenas too short waiting times.

Schmickl and Hamann (Schmickl and Hamann, 2011) defined the waiting time as a sigmoid function

$$w(I) = \frac{w_{\max} I^2}{I^2 + c}, \tag{1}$$

where $w_{\max}$ is a predetermined maximum waiting time in seconds, I is the locally measured luminance, and a constant $c = 4.86 \times 10^5$. The swarm's clustering performance depends on the constants $w_{\max}$ and c. If they are set incorrectly, then robots do not cluster at all or cluster at undesired places. Similarly, these values need to be adapted dynamically in the case of a dynamic light gradient or dynamic swarm densities.

In a previous study, we have introduced an approach to automatically adapt the waiting function to any light configuration by setting the maximal waiting time $w_{\max}$ to a big value which is then scaled according to the minimum light intensity $I_{\min}$ and maximum light intensity $I_{\max}$ measured by the robot in the arena (Wahby et al., 2016). This information is then used to rescale the light intensity $\bar{I}$ to the interval $[0, 1]$ by

$$\bar{I} = \frac{I - I_{\min}}{I_{\max} - I_{\min}}. \tag{2}$$

An additional initial calibration phase was required, where the robots explore the arena and collect illumination samples in regular intervals. For the used arena dimensions (3×2.5 m^2), a calibration period of three minutes was found to

be sufficient for estimating I_{min} and I_{max}. Then, the robots adapt their waiting time individually using $\bar{I}$ instead of I in eq. 1 and start the aggregation phase. This approach succeeds in the automatic adaption to surrounding light conditions. However, if light conditions change during the productive phase of the experiment, then the robot swarm may fail to cluster at the brightest spot. Also, without communication each robot has its own estimation of the light conditions that may lead to inconsistent waiting times and aggregation behavior within the swarm.

Boosting BEECLUST for Dynamic Swarm Densities and Dynamic Environments

We extend the BEECLUST algorithm by making the robots aware of the swarm density and by allowing for more collaboration between robots in order to share measurements (see Fig. 1). When a robot moves, it measures the average time taken between robot-to-robot-encounters t_r, to get an estimate of the swarm density. Instead of a calibration phase, the robots collect measurements of the light during the actual execution of the experiment. Each robot measures and stores a light intensity sample every second. Robots share features of these measurements ('communicate parameters', Fig. 1) to collectively perceive the dynamic environment and to adapt their waiting times accordingly. Not every robot has to travel individually to all relevant places in the arena to collect appropriate measurements.

According to kinetic gas theory, the mean free path $\ell = (n\sigma)^{-1}$ is the average distance traveled by a moving particle between successive collisions with other moving particles. n is the number density of molecules (ideal gas law) and $\sigma = \pi(2r)^2$ is the effective cross-sectional area for spherical particles with radius r. This indicates that, in swarm robotics, the density of moving robots on a fixed area may scale similarly, however, inversely proportional to the average robot-to-robot-encounter time interval t_r (Kernbach et al., 2013). For example, if we halve the swarm size, then the robots measure on average a 100% increase of time t_r, that is, they should also increase their waiting time by 100% for effective clustering.

Therefore, we include the average time between two robot encounters t_r into the waiting function as a product

$$w(\bar{I}) = \frac{t_r}{12}\,\frac{w_{\text{max}}\bar{I}^2}{\bar{I}^2 + c}\,, \tag{3}$$

which substitutes the original waiting function given in eq. 1. We have to introduce another constant here as we normalize t_r by 12. This corresponds to the average value measured for t_r (in seconds) in our arena. By normalizing t_r we can keep the value of the maximal waiting time w_{max} as it is and preserve its interpretability.

Robots share their measured values by communication with other swarm members to ensure the consistency of their behaviors and to enhance the swarm's collective performance. Robots locally exchange their adaptivity parameters (I_{min}, I_{max}, and t_r) at every time they are in proximity of other robots. After receiving parameters from a neighbor, a robot checks the new I_{min} and I_{max} and computes a new average t_r. According to the robots' *memory lifetime*, the measurements are considered outdated after a certain amount of time (90 seconds). The *memory lifetime* introduces a trade-off between the adaptivity and speed.

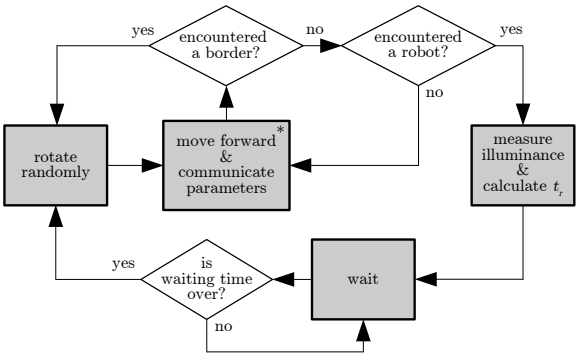


Figure 1: Finite state machine describing the adaptive BEECLUST algorithm.

Robot and Experiment Setup

Thymio II is a small mobile robot designed for children (Riedo et al., 2013), it measures approximately $11 \times 11 \times 5$ cm^3 in width, depth, and height. Its programming is based on ASEBA (Magnenat et al., 2011), which is an open source modular architecture for event-based robot control. The robot is based on a differential wheeled drive, and is equipped with many sensors, such as seven horizontal infrared sensors, two ground infrared sensors, a 3-axis accelerometer, a remote control infrared receiver, and a temperature sensor. Thymio II robots also provide attachment knobs for Lego$^{\text{TM}}$ bricks on top, which allows an easy installation of hardware extensions. The infrared sensors are used for kin recognition, ensuring a robot makes a transition to the stopped state only when another robot is detected at a distance of 5 cm or less. A second purpose of the infrared sensors is robot-to-robot communication. Two Thymios can communicate over distances of up to about 25 cm, as long as at least one of the seven IR emitters is within line of sight of a horizontal sensor of the other robot. For bi-directional communication this alignment has to be reciprocal.

In our experiments, the robots need to communicate the extremes of their memorized light intensity measurements I_{min} and I_{max} as well as their swarm density estimation parameter t_r. However, the angular displacement of the horizontal sensors does not provide a full circumferential view. The sides of a Thymio robot have no infrared sensors, (see Fig. 2) leaving two large blind areas without

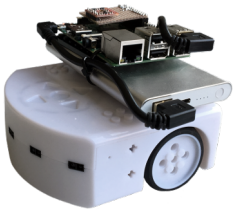


Figure 2: Thymio II robot with a Raspberry Pi, an ambient light sensor, and a power bank attached to the Lego™ attachment knobs on top of the robot.

possible communication. To increase the chances for communication, we rotate the robots on the spot while in the 'stopped state' which vastly increases frequency of successful communications of waiting robots. Thymios can send 11 bit messages at 100 ms intervals, with one bit being used by the firmware itself leaving 10 bits for user data. The first two bits determine the type of the communicated parameter (00 for I_{min}, 01 for I_{max}, and 10 for t_r), and the parameter value is encoded in the remaining eight bits as integer on the interval $[0, 255]$. In the case a robot receives two subsequent messages of the same type and value, the earlier message is discarded to avoid duplicates. The robots don't relay received messages and delete them after the *memory lifetime* has been reached. This is an important feature that ensures robustness because it prevents spreading of faulty or outdated information (Schmickl et al., 2006). It also acts as first line of defense against Byzantine robots.

According to our experiment setup, the robots go through different phases of light intensity configurations (see Fig. 4) of at least six minutes each to simulate a dynamic environment. Therefore, we set the *memory lifetime* to 90 seconds, which is a good compromise between quick adaptation to such quick changes and still relying on averaged measurements. Although our motivation is to maximize the degree of adaptivity in our system, a few parameters, such as as the *memory lifetime* and the normalization of t_r by 12 in Eq. 1, are required. Certainly, shorter *memory lifetimes* may be advantageous in even more dynamic environments and longer intervals may help in environments that change slowly. However, we assume that these environmental features are unknown a priori and cannot be anticipated.

Following our earlier approach (Wahby et al., 2016), we extend the capabilities of the Thymio II robots by a hardware extension consisting of a Raspberry Pi, an ambient light sensor (TSL45315), a power bank, and a Lego™ attachment plate (see Fig. 2). The ambient light sensor is interfaced with the Raspberry Pi through the I^2C bus and it provides an output range that corresponds to environmental conditions ranging from 3 lux to 2.2×10^5 lux. The Raspberry Pi connects to the Thymio II through the USB port and the D-Bus

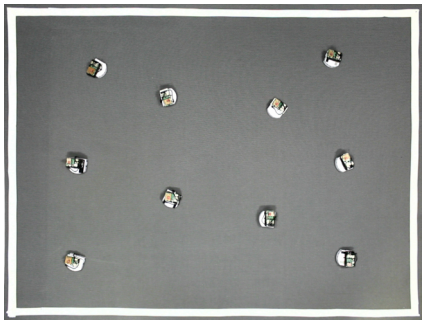


Figure 3: The 3.0×2.5 m^2 arena is bound by a white line on the ground (duct tape) emulating virtual walls that can be detected by the robots' ground sensors. $N = 10$ robots are initially placed in an evenly spaced grid.

interface, and considers the robot as a sensor/actuator unit[1].

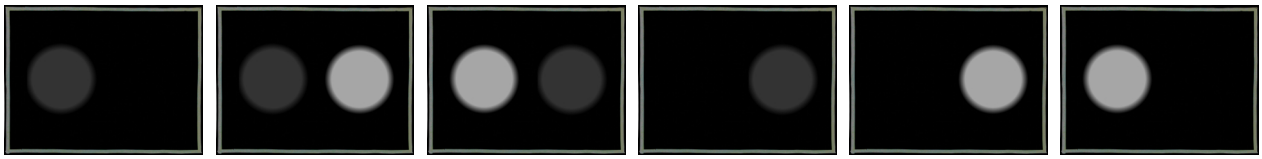
In order to have comparability to previous studies based on other versions of the BEECLUST algorithm as in (Wahby et al., 2016), (Schmickl et al., 2008), and (Schmickl et al., 2009), we constructed an arena of size 3.0×2.5 m^2 (see Fig. 3). This arena is bounded by a white line on the ground emulating virtual walls that can be detected by the robots' ground sensors. In the case a ground sensor detects a virtual wall, the robot rotates into the opposite direction for a random amount of time between 0.5 and 2.2 seconds, before it continues moving forward. The virtual walls allow for using the horizontal proximity sensing exclusively for kin recognition, which increases the reliability.

When moving forward, the robots operate in full speed of 20 cm/second. Two projectors are fixed to the ceiling above the arena, that allows the projection of two light spots (each at the middle of one half of the arena), in different intensity configurations (see Fig. 4). With this setup we can demonstrate different aggregation behaviours while keeping the configuration simple and reproducible. At the beginning of each experiment, ten robots are initially placed in an evenly spaced grid (see Fig. 3). All the experiments are video captured by a digital camera, which is also fixed to the ceiling. We divided each experiment into non-overlapping intervals of five seconds. For every interval, we counted the maximum number of clustered robots (i.e., in the stopped state) at each of the two light spots. Then the median, minimum and maximum values at every four intervals (from $n = 5$ repetitions) are used to construct the graphs in Figs. 5 and 6.

Results

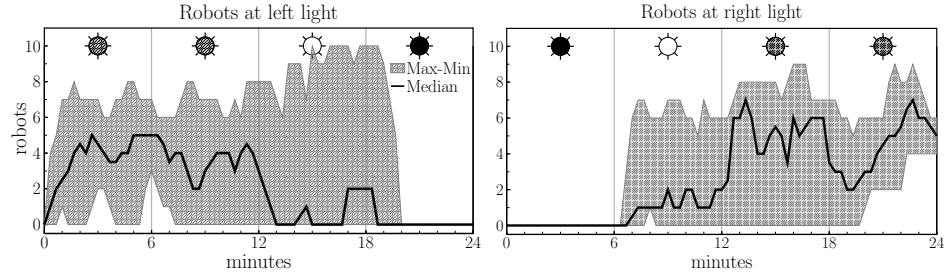
We tested four variants of the BEECLUST algorithm: *offline* (with initial calibration but no online illumination sampling and no communication, similar to our previous approach in (Wahby et al., 2016)), *online non-communicating*

[1] source code:
https://git.iti.uni-luebeck.de/CEschke/A_BEE.git

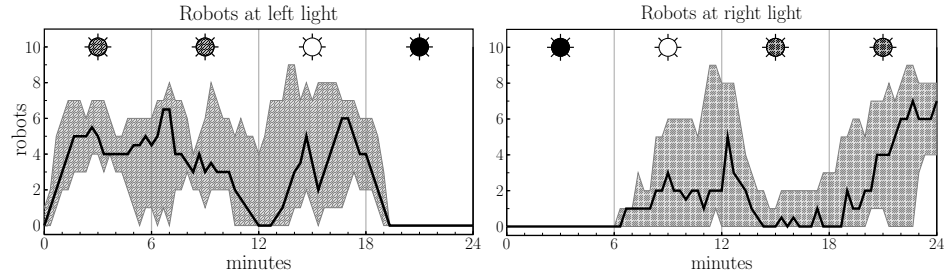


(a) dimmed / off (b) dimmed / bright (c) bright / dimmed (d) off / dimmed (e) off / bright (f) bright / off

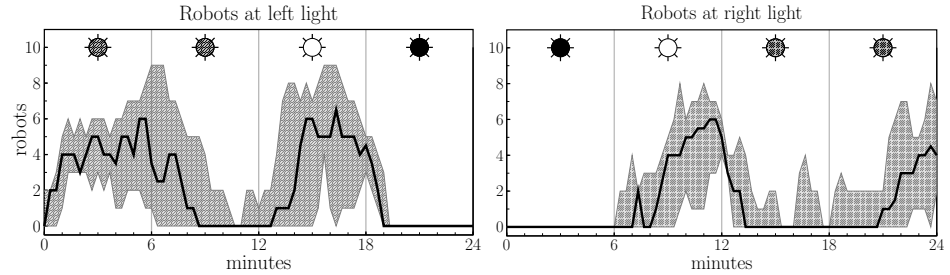
Figure 4: Light configurations of the arena during the different phases of the two kinds of experiments. In the light adaptivity experiments (our first three sets of experiments) the light configuration sequence (a) → (b) → (c) → (d) is used. For the swarm density adaptivity experiments (the last two set of experiments) (e) followed by (f) is used. The projector operates at 100% brightness to project bright spots, which is measured as about 30 lux by the light sensor of our robots. 20% brightness is projected for dimmed spots, which is measures as about six lux. Both light spots are about 70 cm in diameter.



(a) offline approach (initial calibration phase, no online illumination sampling and no communication; control experiment).



(b) online non-communicating approach (online illumination sampling but no communication).



(c) online communicating without density estimation approach (with online illumination sampling, swarm density sampling, and communication).

Figure 5: Static density runs, analysis of practical robot experiments using $N = 10$ Thymio robots and the different variants of the BEECLUST algorithm in consecutive phases of light settings. Each of the five rows represents the results of $n = 5$ repetitions and gives the number of aggregated robots at the left and right area. The shown data is obtained by processing the time series in 20 second segments, extracting the maximum, minimum, and median number of robots observed over all $n = 5$ repetitions within that segment. The sun-like symbols indicate the dynamically changing light intensity at the corresponding projected light spot. A white sun indicates a bright light spot, a gray sun indicates a dimmed light spot, and a black sun indicates no light.

(with online illumination sampling but no communication), *online communicating without density estimation* (with online illumination sampling and communication), and *online communicating with density estimation* (with online illumination sampling, swarm density sampling, and communication). The first three BEECLUST variants were tested in a first experiment setup with dynamic light conditions and static density. There are four different consecutive phases of light settings, six minutes each (see Fig. 4). In the second experiment setup we tested the adaptivity to dynamic swarm densities using the *online communicating without density estimation* BEECLUST variant as control experiment and the *online communicating with density estimation* BEECLUST variant. This second experiment setup consists of only two phases with a slightly longer duration of 14 minutes each, where the swarm is manually halved from $N = 10$ robots to $N = 5$ robots after 14 minutes. Each experiment is repeated $n = 5$ times (totaling to 25 robot experiments)[2]. We test for statistical significance by comparing numbers of clustered robots pooled over the whole phase from all $n = 5$ repetitions using the Wilcoxon signed-rank test ($p \leq 0.001$).

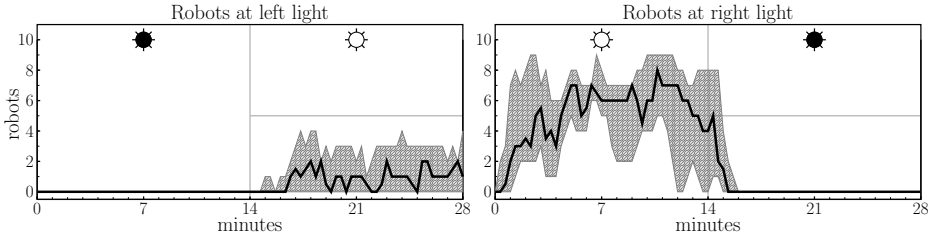
In Fig. 5(a) we give the results for the *offline* BEECLUST variant as control experiment. Before the experiment starts, the robots adapt their waiting time individually to the light configuration in the first phase (*dimmed / off*). Here, the robots learn that the dimmed spot is the brightest possible spot, and in case they encounter a brighter spot in the later phases, the maximum waiting time computed during the initial calibration is still considered. Focusing on the medians given in Fig. 5(a), we notice that during the first two minutes in the first phase, the number of robots clustering at the dimmed spot increases about linearly. Then a median of about five is reached and only minor fluctuations occur due to scouting behaviors until the end of the phase. In the next phase (*dimmed / bright*), a bright spot emerges at the right side of the arena. However, clusters of only two robots form at the bright spot, because the robots are assured by the initial calibration phase that they are at the brightest spot already. In this and the third phase, following the minimum, maximum, and median values of clustering robots, it is obvious that the robots are unable to locate the brighter spot until the bright spot disappears in the last phase.

In Fig. 5(b) we give the results for the *online non-communicating without density estimation* variant, that is, robots do not require an initial offline calibration phase but sample light measurements online. Similar to the above experiment, the robots form a cluster at the dimmed spot during the first phase. When the bright spot emerges at the right side of the arena, the cluster of about five robots at the dimmed spot required almost the whole duration of the phase to dissolve and to form again at the bright spot. Possibly robots are assured by locally collected information that they are at a bright spot already, while the scouting robots fail to communicate the information about the new bright spot. From the results of the remaining two phases it is obvious that this variant outperforms the *offline* approach in terms of adapting to dynamic light conditions (significantly more clustered robots during both phases, $p \leq 0.001$).It deals relatively well with a required aggregation at the dimmed spots but fails to aggregate many robots quickly at the bright spots (second phase, right; third phase, left).

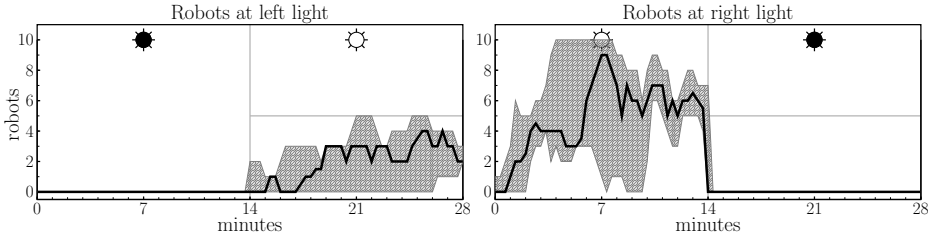
In Fig. 5(c) we show the results for testing the *online communicating without density estimation* variant, that is, we test whether we can improve performance once we allow the robots to communicate their light adaptivity parameters (I_{min} and I_{max}). Similar to the previous two setups, the robots form a cluster at the dimmed spot during the first phase. When the bright spot emerges at the right side of the arena, the cluster at the left side quickly dissolves and the robots form a new cluster at the bright spot. The scouting robots probably detect the bright spot in the arena and collect light intensity samples of bigger values (i.e., acquired high I_{max}). They successfully share their knowledge with the robots clustered at the dimmed spot. This way the aggregated robots notice that they are currently clustering at a local optimum without having to explore the arena themselves. This allows to quickly dissolve the cluster. We find that this variant is performing significantly better than the *online non-communicating without density estimation* variant when bright light spots emerge in new locations. In the third phase, the cluster at the dimmed spot quickly dissolves and forms again at the bright spot. In the last phase, the dimmed spot is at the right side of the arena. Since the robots can retain an I_{max} value for up to three minutes after it was acquired by another robot and then communicated, the first half of this phase shows no reliable cluster formation. Probably the waiting times of the robots are still scaled for an optimum with a bright spot and not a dimmed spot. During the last three minutes of the experiment, however, they form a cluster at the right side. Especially the performance of the *online communicating without density estimation* variant during phase two and three is a clear improvement over the *online non-communicating without density estimation* variant. The performance increase is statistically significant (Wilcoxon signed-rank test, $p \leq 0.001$).

Finally, we test our BEECLUST approach in a setting with dynamic swarm densities. We compare the performance of our *online communicating with density estimation* BEECLUST approach, see Fig. 6(b), to a control experiment using the *online communicating without density estimation* variant, see Fig. 6(a). In the first phase (*off / bright*) we start with $N = 10$ robots. Big robot clusters of up to nine or ten robots are formed at the bright spot. At the beginning of the second phase, the bright spot is switched to the left side and half of the robot swarm (five robots) is randomly manually removed from the arena, only $N = 5$ robots remain. The

[2] video available online: https://vimeo.com/271398596



(a) online communicating without density estimation approach (control experiment).



(b) online communicating with density estimation approach

Figure 6: Dynamic density runs, analysis of practical robot experiments using initially $N = 10$ Thymio robots, then the swarm size is manually reduced to $N = 5$ robots at $t = 14$ min. To identify the advantage of the density estimation approach, only the 2nd phase ($14 < t \leq 28$ min) is relevant. Significantly more robots aggregate at the left light for $t > 14$ min using density estimation. $n = 5$ repetitions per row, number of aggregated robots at the left and right area, dynamically changing light intensity.

clusters quickly dissolve as usual due to the shared knowledge from scouts. The *online communicating with density estimation* variant succeeds in forming robot clusters of up to five robots (i.e., the complete swarm) at the bright spot after about seven minutes. The *online communicating with density estimation* variant outperforms the control experiment because significantly more robots cluster during the second phase of the experiment ($p < 0.001$). In these two experiments light intensity was not changed, hence, clustering at the opposite side of the arena was not delayed due to the robots' local history. The observed behavior in this setup indicates the ability of our approach to adapt to the dynamic change in the swarm size and successfully aggregate the swarm at the desired area. We have analyzed the measured robot-to-robot encounter time t_r of an arbitrary robot. In the first phase, t_r is about 13 seconds in average. In the second phase, it increases to about 18 seconds in average. Hence, the robots successfully detect the change in swarm density and adapt their behavior. All experiment data, photos, and videos are available online (Wahby et al., 2019).

Discussion and Conclusion

We have extended the BEECLUST algorithm with a method to adapt waiting times to dynamic swarm densities. Times between robot-to-robot encounters and environmental features (information about the light distribution) are measured and gathered by each robot, shared by communication, and aggregated. This way we contribute a solution to the challenge of collectively detecting global features and their changes, that is, collective change detection. We also found parameters to balance the tradeoff between minimal detection delay and false alarms. We have successfully verified our approach in 25 swarm robot experiments that show a clear improvement in performance over the non-adaptive variant. Adaptivity to swarm density is a relevant feature in swarm robotics with possible impact on the swarm's scalability properties. Similarly, adaptivity to environmental features is crucial for the operation in dynamic environments. Both properties enable long term autonomy and are essential in real world applications (e.g., environmental monitoring).

An interesting extension in future work is to combine collective perception (Schmickl et al., 2007) with other scenarios of swarm robotics to allow for adaptivity to even more features. Other options for future work include a more intensive study of adaptation to swarm density and scalability. A better control of when data shared with other robots is outdated could be achieved by adding time stamps. However, the Thymio's infrared-based communication may be too primitive and WiFi or scalable protocols could be used instead. Finally, we could try to protect our adaptivity process against malicious robots that intentionally or by accident spread wrong measurements. Methods of error detection and correction in distributed robot systems may be useful (Tarapore et al., 2015).

References

Arvin, F., Turgut, A. E., Bazyari, F., Arikan, K. B., Bellotto, N., and Yue, S. (2014). Cue-based aggregation with a mobile

robot swarm: a novel fuzzy-based method. *Adaptive Behavior*, 22(3):189–206.

Arvin, F., Turgut, A. E., and Yue, S. (2012). Fuzzy-Based Aggregation with a Mobile Robot Swarm. In *Swarm Intelligence (ANTS'12)*, volume 7461 of *Lecture Notes in Computer Science*, pages 346–347, Berlin. Springer.

Bonabeau, E., Dorigo, M., and Theraulaz, G. (1999). *Swarm Intelligence: From Natural to Artificial Systems*. Oxford Univ. Press, New York.

Correll, N. and Hamann, H. (2015). Probabilistic Modeling of Swarming Systems. In Kacprzyk, J. and Pedrycz, W., editors, *Springer Handbook of Computational Intelligence*, pages 1423–1431. Springer.

Hamann, H. (2018). *Swarm Robotics: A Formal Approach*. Springer.

Kernbach, S., Häbe, D., Kernbach, O., Thenius, R., Radspieler, G., Kimura, T., and Schmickl, T. (2013). Adaptive collective decision-making in limited robot swarms without communication. *The International Journal of Robotics Research*, 32(1):35–55.

Kernbach, S., Thenius, R., Kornienko, O., and Schmickl, T. (2009). Re-embodiment of Honeybee Aggregation Behavior in an Artificial Micro-Robotic Swarm. *Adaptive Behavior*, 17:237–259.

Khaluf, Y., Pinciroli, C., Valentini, G., and Hamann, H. (2017). The Impact of Agent Density on Scalability in Collective Systems: Noise-Induced versus Majority-Based Bistability. *Swarm Intelligence*, 11(2):155–179.

Magnenat, S., Rétornaz, P., Bonani, M., Longchamp, V., and Mondada, F. (2011). ASEBA: A Modular Architecture for Event-Based Control of Complex Robots. *IEEE/ASME transactions on mechatronics*, 16(2):321–329.

Mallon, E. B., Pratt, S. C., and Franks, N. R. (2001). Individual and Collective Decision-Making During Nest Site Selection by the Ant *Leptothorax albipennis*. *Behavioral Ecology and Sociobiology*, 50:352–359.

Mayya, S., Pierpaoli, P., and Egerstedt, M. (2018). Voluntary retreat for decentralized interference reduction in robot swarms. *arXiv preprint arXiv:1812.02193*.

Picard, D. (1985). Testing and estimating change-points in time series. *Advances in Applied Probability*, 17(4):841–867.

Riedo, F., Chevalier, M., Magnenat, S., and Mondada, F. (2013). Thymio II, a Robot that Grows Wiser with Children. In *Advanced Robotics and its Social Impacts (ARSO), 2013 IEEE Workshop on*, pages 187–193. IEEE.

Schmickl, T. and Hamann, H. (2011). BEECLUST: A Swarm Algorithm Derived from Honeybees. In Xiao, Y., editor, *Bio-inspired Computing and Communication Networks*. CRC Press.

Schmickl, T., Hamann, H., Wörn, H., and Crailsheim, K. (2009). Two Different Approaches to a Macroscopic Model of a Bio-Inspired Robotic Swarm. *Robotics and Autonomous Systems*, 57(9):913–921.

Schmickl, T., Möslinger, C., and Crailsheim, K. (2006). Collective perception in a robot swarm. In *International Workshop on Swarm Robotics*, pages 144–157. Springer.

Schmickl, T., Möslinger, C., and Crailsheim, K. (2007). Collective Perception in a Robot Swarm. In Şahin, E., Spears, W. M., and Winfield, A. F. T., editors, *Swarm Robotics - Second SAB 2006 International Workshop*, volume 4433 of *Lecture Notes in Computer Science*, Heidelberg/Berlin, Germany. Springer-Verlag.

Schmickl, T., Thenius, R., Möslinger, C., Radspieler, G., Kernbach, S., and Crailsheim, K. (2008). Get in touch: cooperative decision making Based on robot-to-robot collisions. *Autonomous Agents and Multi-Agent Systems*, 18(1):133–155.

Soysal, O. and Şahin, E. (2007). A Macroscopic Model for Self-organized Aggregation in Swarm Robotic Systems. In Şahin, E., Spears, W. M., and Winfield, A. F. T., editors, *Swarm Robotics - Second SAB 2006 International Workshop*, volume 4433 of *Lecture Notes in Computer Science*, pages 27–42, Berlin, Germany. Springer-Verlag.

Szopek, M., Schmickl, T., Thenius, R., Radspieler, G., and Crailsheim, K. (2013). Dynamics of Collective Decision Making of Honeybees in Complex Temperature Fields. *PLoS ONE*, 8(10):e76250.

Tarapore, D., Lima, P. U., Carneiro, J., and Christensen, A. L. (2015). To err is Robotic, to Tolerate Immunological: Fault Detection in Multirobot Systems. *Bioinspiration & biomimetics*, 10(1):016014.

Trianni, V., Groß, R., Labella, T. H., Şahin, E., and Dorigo, M. (2003). Evolving Aggregation Behaviors in a Swarm of Robots. *Lecture Notes in Artifcial Intelligence*, 2801:865–874.

Valentini, G., Brambilla, D., Hamann, H., and Dorigo, M. (2016). Collective Perception of Environmental Features in a Robot Swarm. In Dorigo, M., Birattari, M., Li, X., López-Ibáñez, M., Ohkura, K., Pinciroli, C., and Stützle, T., editors, *Swarm Intelligence*, pages 65–76, Cham. Springer International Publishing.

Valentini, G. and Hamann, H. (2015). Time-Variant Feedback Processes in Collective Decision-Making Systems: Influence and Effect of Dynamic Neighborhood Sizes. *Swarm Intelligence*, 9(2-3):153–176.

Wahby, M., Petzold, J., Eschke, C., Schmickl, T., and Hamann, H. (2019). Supplementary material online. https://doi.org/10.5281/zenodo.1419277.

Wahby, M., Weinhold, A., and Hamann, H. (2016). Revisiting beeclust: Aggregation of swarm robots with adaptiveness to different light settings. *EAI Endorsed Transactions on Collaborative Computing*, 2(9).