Collective Decision-Making and Change Detection with Bayesian Robots in Dynamic Environments

Kai Pfister[1] and Heiko Hamann[2]

***Abstract*— Solving complex problems collectively with simple entities is a challenging task for swarm robotics. For the task of collective decision-making, robots decide based on local observations on the microscopic level to achieve consensus on the macroscopic level. We study this problem for a common benchmark of classifying distributed features in a binary dynamic environment. Our special focus is on environmental features that are dynamic as they change during the experiment. We present a control algorithm that uses sophisticated statistical change detection in combination with Bayesian robots to classify dynamic environments. The main profit is to reduce false positives allowing for improved speed and accuracy in decision-making. Supported by results from various simulated experiments, we introduce three feedback loops to balance speed and accuracy. In our benchmarks, we show the superiority of our new approach over previous works on Bayesian robots. Our approach of using change detection shows a more reliable detection of environmental changes. This enables the swarm to successfully classify even difficult environments (i.e., hard to detect differences between the binary features), while achieving faster and more accurate results in simpler environments.**

I. Introduction

Solving complex problems with many rather simple robots is a key idea of swarm robotics [1]. A typical challenge in swarm robotics is to conceptually connect the robot controller operating on the microscopic level of local perception and neighbor-to-neighbor communication with the requirements for the whole swarm on the macroscopic level. Collective decision-making (CDM) is an important skill that helps to establish a micro-macro-link and enables the swarm as a whole to act autonomously [2]. Here, we focus on the specific challenge making collective decisions in dynamic environments. In the real world a robot's environment is never completely static. Changes constantly happen all the time in a robot's surroundings (e.g., light conditions), due to other robots being around, or due to the robot's own actions. This type of adaptivity in swarm robotics is becoming an increasingly important field of study in CDM, pushing previous approaches to new limits.

In a previous simulation study, we investigated collective decision making (CDM) of so-called 'Bayesian robots' in dynamic environments with spatially distributed binary features [3]. We enabled the static 'Bayes Bots' by Ebert et al. [4] to change their initial decision. For this purpose, the Bayesian robots were equipped with a threshold that triggers the resetting of the robots' belief state. With a robot swarm capable of responding to changes in the environment at our hands, we also studied the major challenge of collective decision-making: the speed-vs-accuracy tradeoff. We showed that a conservative swarm quickly finds a consensus but lacks ability to adapt to changes, especially rapid changes. A progressive swarm instead is quickly able to react to such changes but has difficulties in reaching consensus within the swarm. Noteworthy results for our Dynamic 'Bayes Bots' (DBB) [3] were an increase of false positives (i.e., robots switching opinion without environmental change or reversing their correct decision) and an overall decrease in accuracy in environments that are particularly difficult.

Here, we try to further increase the capabilities of robot swarms to collectively, quickly, and accurately adapt to environmental changes. Our key concept is to apply methods of statistical change detection in our distributed robot system. Change detection is an established field in statistical analysis of time series. The main contribution of this paper is the selection of appropriate methods from change detection, their adaptation to our strictly decentralized distributed robot system, and the analysis in swarm simulations.

II. Related Work

A. Collective Decision-making

CDM is essential in swarm robotics and has various areas of application. It is the art of multiple robots forming a consensus by aggregating local measurements and beliefs to quickly gain an accurate global picture. Many aspects have been addressed over the last decade, for example, multi-feature environments [2], [5], the influence of malicious agents [6], [7], the crucial speed-vs-accuracy tradeoff [4], [3], and the exploration of options [8].

Most of the literature of CDM focuses on benchmarks, such as nest-site selection (best-of-n) or collective classification tasks in static environments [4], [9], [10]. A popular benchmarking task is the so-called collective perception scenario, an evaluation of a two-dimensional environment consisting of black and white tiles. In this binary discrimination problem, the swarm must decide on which tile color is dominant. Ebert et al. [4] investigated the unavoidable tradeoff between speed and accuracy in such a binary classification problem. A parameter sweep was performed to analyze the best performing parameter constellations for Bayesian robots within environments of varying difficulty. Similar to the work of Bartashevich and Mostaghim [11], Ebert et al. [4] showed the influence of spatial correlations between observations (e.g., patches of black tiles). The robots estimate the ratio of black and white tiles by a model-based

[1]KP is at the Institute of Computer Engineering, University of Luebeck, Germany; [2]HH is at the Department of Computer and Information Science, University of Konstanz, Germany heiko.hamann@uni-konstanz.de

approach that forms a belief based on a beta distribution. This allows the collective to form an accurate estimate of whether the environment is mostly black or white without any prior knowledge.

Another form of Bayesian hypothesis testing in binary CDM is presented by Shan and Mostaghim [10]. The Distributed Bayesian Hypothesis Testing is not fully distributed. After a distributed collection of samples by the individual agents, the observations are merged in a centralized opinion fusion process, self-organized by a chosen leader.

Prasetyo et al. [12] studied change by introducing dynamic nest sites (nest site qualities can change abruptly). They showed an improvement of the swarm's adaptability with a small number of stubborn agents that keep exploring their initial opinion independently. The switch to another opinion is probability-based and independent of the prior decision-making process. A similar scenario was studied by Talamali et al. [13]. The task difficulty is increased by changing the number of available nest sites and their quality during the trial. They studied the influence of a robot's communication range on the adaptability of the swarm. They show the suprising result that decreasing the communication range can favor the local distribution of information. A similar study by Aust et al. [14] confirmed these findings and found a similar effect concerning the frequency of measurements. Divband Soorati et al. [15] studied dynamic environments. In binary decision-making, evidence is simply counted by robots and a reset mechanism removes outdated measurements.

As mentioned above, we implemented a reset mechanism and three kinds of feedback to avoid consensus lock-ins and improve the adaptability to dynamic environments [3]. This previous method was the starting point for our work here.

B. Change Detection

The analysis for changes within time series is an important and widely researched topic [16], [17]. Analysis of change is important in many different fields, such as medical diagnosis and stock market analysis. Change point detection (CPD) aims at identifying abrupt changes in the statistical properties of the data [16]. CPD is separated into two main scenarios: online and offline. In offline CPD all data is available. Offline techniques detect changes retrospectively and segment the collected data [18]. In online CPD the goal is to detect the change points in 'real time' requiring fast algorithms [19]. Methods of CPD can be clustered in three segments: supervised, unsupervised and semi-supervised [17].

Supervised approaches learn a model that maps input data to predefined classes based on offline training data [20]. For CPD these classes mostly define whether there is a change point. There are supervised approaches for binary and multi-class classifiers [16]. Classes that were not presented in the training data cannot be distinguished [17].

Unsupervised methods come in a great variety and are popular because they do not need specific prior training on labeled data to classify changes [16]. The discovery of patterns in unlabeled data is used to detect anomalies in generative properties of the data [17]. The big advantage of unsupervised methods is that they can be applied to many different cases without the need of intensive prior training.

A general approach for unsupervised CPD is the evaluation of change points based on the likelihood-ratio of two consecutive segments within the data. These methods calculate either the probability density and density ratio or the direct density ratio [21]. The direct density ratio is used in approaches, such as the Kullback-Leibler Importance Estimation Procedure (KLIEP) [22], the Unconstrained Least-Squares Importance Fitting (uLSIF) [23] and the Relative Unconstrained Least-Squares Importance Fitting (RuLSIF) [24]. Approaches that estimate the densities first are the Cumulative Sum (CUSUM) [25] and Change Finder [26].

Other strategies are predictive and probability-based. They use Gaussian processes [27] or Bayesian techniques [28] to estimate change points. An online Bayesian approach is the Bayesian Online Changepoint Detection (BOCPD) [28].

Semi-supervised techniques can be found in recent works but are not widely used for change detection [17]. An advantage over solely supervised methods is the possibility of using a combination of unlabeled and labeled data.

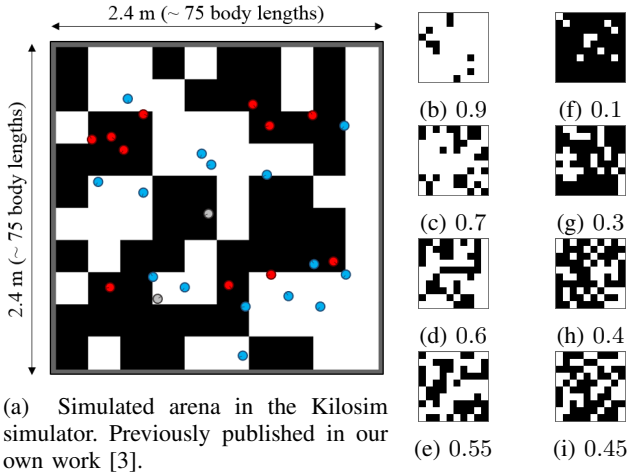
C. Pruned Exact Linear Time (PELT)

Killick et al. [29] introduced Pruned Exact Linear Time (PELT). It is a fast, accurate and unsupervised method for CPD. PELT sets out to find the optimal number and location of change points while maintaining linear computational cost under specific conditions. Therefore, they extended the Optimal Partitioning (OP) method by Jackson et al. [30] with a pruning step that frees the dynamic program of non-relevant change point locations in future computations. The pruning operation reduces computational costs and maintains the accuracy of the segmentation. In a preliminary method selection (data not shown), we found that PELT outperformed methods, such as the Bayesian Online Changepoint Detection [28] and Relative Unconstrained Least-Squares Importance Fitting [21]. In our scenario, PELT is best in terms of speed, accuracy, and robustness. Hence, we selected PELT as our CPD method of choice.

III. Scenario and Simulation

The overall task of the simulated collective is to solve a typical binary best-of-n problem (n = 2). We investigate the behavior of a robot swarm of size $N = 100$ in a dynamic environment. The environment is defined as a two-dimensional arena consisting of black and white tiles. The ratio of black to white tiles inverts at halftime, setting a new best option. This change forms the dynamic environment. In Fig. 1a the simulated arena is shown. It is two dimensional and represents a bordered area of 2.4 m $\times$ 2.4 m.

All of our experiments are simulated using the Kilosim simulator [31]. The simulated robot platform is the Kilobot. The lighting in the arena of the Kilosim is simulated by the so-called light-pattern, which is a grid of ten by ten randomly arranged black and white tiles. When the white tiles are in the majority the environment is predominantly bright. The difficulty of the scenario is defined by the fill ratio of black



(a) Simulated arena in the Kilosim simulator. Previously published in our own work [3].

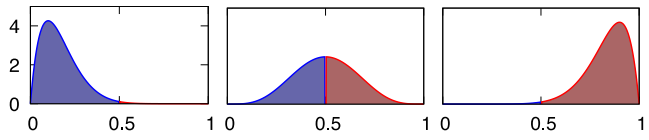
Fig. 1: Simulated robot arena and tiling patterns (similar to [4]). (a) Top view of robot arena with 10-by-10 black and white tiling pattern; robots as circles with color-code: in favor of white (blue), in favor of black (red), and uncommitted (gray). (b, c, ..., i) Randomly generated tilings for task difficulties $f \in \{0.9, 0.7, 0.6, 0.55, 0.45, 0.4, 0.3, 0.1\}$ (ratio of white vs black tiles).

and white tiles in the arena. A fill ratio of 1.0 defines a completely white environment and a fill ratio of 0.0 defines a completely black one (fill ratio = 1-(black tiles/100)). The closer the fill ratio is to 0.5 the harder it is for the collective to distinguish which color is in the majority. To simulate dynamic environments the light-pattern changes abruptly to the opposite fill ratio (e.g., fill ratio of 0.9 changes to 0.1) at the half-time of the trial duration.

The $N = 100$ simulated Kilobots are initially positioned in a regular grid-pattern close to a uniform distribution. The swarm has 5,000 seconds before the switch to the opposite fill ratio and 5,000 seconds afterwards to classify each environment, resulting in a total trial duration of 10,000 seconds. For each scenario we do 20 independent runs. All trials start with a dominantly white environment which then transitions to a mostly black environment, as shown in Figs. 1 (b) to (i).

IV. Approach

The robots observe the environment continuously and detect changes in the environment applying a sophisticated change detection method. The fill ratio of the environment is mapped by each robot as a belief based on its local observations (i.e., evidence). Robots keep collecting evidence while already making decisions. Based on our previous work [3], our goal is to improve reaction times and accuracy in dynamic environments. Using the sophisticated state-of-the-art change detection method PELT we hope to reduce false positives and false negatives (i.e., maximizing sensitivity and specificity). If a change is detected, robots initiate a reset and restart collecting evidence (details below). A robot is in either of 3 states: uncommitted, in favor of black, or in favor of white. Initially robots are uncommitted. Later they commit to whether there is a majority of black tiles or white tiles. We test new feedback mechanisms adjusted to applications of change detection.



(a) $\alpha = 2, \beta = 10$ (b) $\alpha = 5, \beta = 5$ (c) $\alpha = 10, \beta = 2$

Fig. 2: Beta distributions for example parameter settings: (a) majority of black tiles, (b) fifty/fifty, and (c) majority of white tiles. Previously published in our own work [3].

Each robot commits to a decision once the robot reaches a credible threshold of either 0.95 or 0.05 with its belief (1 for white, 0 for black). A robot returns to uncommitted if it detects a change or is convinced by neighbors. We monitor a 'swarm belief' by summing all robot beliefs divided by swarm size N. We define decision-making accuracy. If the swarm belief reaches 1.0 in a mostly white environment, we have perfect accuracy while beliefs of 0.5 or 0.0 are failures. A swarm belief of ≥ 0.8 for predominantly white environments is a satisfactory result (≤ 0.2 for black).

A. Distributed Bayesian Approach

The observational interval τ indicates the time during which the robot collects samples. Increasing the observational interval leads to less collected samples during the run but also to more accurate observations of the environment due to spatial independence of the samples. The robots have limited time to classify the environment before the change and need a sufficient number of samples to do so. By conducting a few test trials prior to the recorded experiments, an observational interval set to τ = 20 seconds showed promise of a good tradeoff. Within 20 seconds the robot can cross up to around eight tiles in the arena when moving in a straight line before collecting the next sample. The robot's belief of the fill ratio of black and white tiles is formed by the lower cumulative distribution function (CDF) at 0.5 of the Beta distribution. In Fig. 2 we give three examples of how parameters α and β influence the distribution. Here, β models the number of black tiles while α models white tiles. Whether the environment is mostly black or white is decided once a robot has observed enough samples of one color and reaches a credible threshold p_c. The Beta distribution is continuously updated with the new color observations C.

B. Collective Decision-making with Change Point Detection

The combination of three feedback types with PELT (see Sec. II-C) as sophisticated CPD forms our new approach called Dynamic 'Bayes Bots' with sophisticated Change Point Detection (DBBCPD). The following pseudo-code (Alg. 1) gives our new approach of DBBCPD using PELT (Alg. 1, line 37). Using DBB [3] as a basis to classify dynamic environments, the new approach uses a Bayesian swarm that can revise its decision. The robots start observing and moving around in the environment by means of a random walk.

Algorithm 1 Dynamic 'Bayes Bots' with CPD

Input: Observational interval τ, credible threshold p_c, prior parameter α_0, neighbor thresholds n_1, n_2 and n_3
Output: Decision d_f and belief q of all robots at time t

1: Init counter of white/black observ. $\alpha \leftarrow \alpha_0$, $\beta \leftarrow \alpha_0$
2: Init incomplete decision $d_f \leftarrow -1$
3: Init reset flag/reset counter $d_{\text{res}} \leftarrow 0$, $r \leftarrow 0$
4: Init dictionary of observations $s_{\text{obs}} = []$
5: Init dictionary of received reset $s_{res} = []$
6: Init dictionary of received recruitment $s_{\text{recruit}} = []$
7: Init dictionary of received transformation $s_{\text{trans}} = []$
8: **for** $t \in [1, T]$ **do**
9: `Perform pseudo-random walk`
10: `Let` $m = (id', r', d'_{\text{res}}, d'_f, \alpha', \beta')$ ▷ msg. of neighb.
11: **if** $d'_{\text{res}} = 1$ and $d_f \neq -1$ and $r \leq r'$ and id' not in s_{res} **then**
12: **if** $n_1 \geq$ neighbors with active reset flag **then**
13: → `accept reset`
14: $(d_{\text{res}} \leftarrow 1)$
15: **if** $d'_f \neq -1$ and $d_f = -1$ and $r \leq r'$ and id' not in s_{recruit} **then**
16: **if** $n_2 \geq$ neighbors with certain decision **then**
17: →`get recruited to the decision`
18: $(d_f \leftarrow d'_f, \alpha \leftarrow \alpha', \beta \leftarrow \beta', r \leftarrow r')$
19: **if** $d'_f \neq -1$ and $d_f \neq -1$ and id' not in s_{trans} **then**
20: **if** $n_3 \geq$ neighbors for opposite decision **then**
21: →`get transformed by decision`
22: $(d_f \leftarrow d'_f, \alpha \leftarrow \alpha', \beta \leftarrow \beta', r \leftarrow r')$
23: **if** $d_{\text{res}} \neq 0$ **then**
24: **Reset:** $d_{\text{res}}, d_f, \alpha, \beta, s_{\text{obs}}, s_{res}, s_{\text{recruit}}$ and s_{trans}
25: $r \leftarrow r + 1$
26: **if** `observe` τ **then**
27: $C \leftarrow$ `observed color (0,1)`
28: $\alpha \leftarrow \alpha + C$
29: $\beta \leftarrow \beta + (1 - C)$
30: $s_{\text{obs}} \leftarrow C$
31: $q \leftarrow \int_0^{0.5} \text{Beta}(\alpha, \beta)$ ▷ lower CDF at 0.5
32: **if** $d_f = -1$ **then**
33: **if** $q > p_c$ **then**
34: $d_f \leftarrow 0$
35: **else if** $(1 - q) > p_c$ **then**
36: $d_f \leftarrow 1$
37: $d_{\text{res}} \leftarrow \text{PELT}(s_{\text{obs}})$
38: BROADCAST MESSAGE$(id, r, d_f, \alpha, \beta)$

The random walk is based on Ebert et al. [4] and consists of a straight segment followed by a turn. The duration of the straight segment is drawn from an exponential distribution with a mean of 240 seconds. The turn is determined by drawing from a uniform distribution between 0 and 2 π. During their random walk the robots start communicating with close neighbors. They exchange their ID and current robot parameters within a metric neighborhood of up to seven centimeters. Three types of feedback aid in improving the swarm's adaptability to dynamic environments.

a) Reset feedback (lines 11-14, Alg. 1): This feedback allows robots to propagate resets for a distinct time period in the event of a change. A robot needs to receive an active reset flag from more than n_1=3 individual neighbors before commencing with the reset procedure. Only robots that have already made a decision and have a lower reset counter than their neighbor can be affected by this feedback type.

b) Recruitment feedback (lines 15-18): An undecided agent can be recruited for a distinct decision if it meets more than n_2 = 5 individual neighbors with the following criteria within a set time interval: having higher reset counters and having made that distinct decision. The feedback works based on the majority rule. Agents with an opposite decision lower the counter of met neighbors. Therefore, a majority of n_2 = 5 neighbors must be reached in order for the recruitment to take effect. This feedback positively influences the convergence speed within the swarm but naturally increases the number of false positives in hard environments.

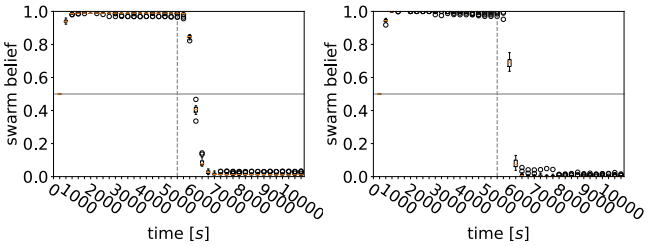
c) Transformation feedback (lines 19-22): This feedback affects only agents that are already committed to a belief. It is similar to the recruitment feedback, however, only addresses robots with opposite beliefs. Committed robots keep monitoring whether they encounter more than n_3 = 10 individual neighbors with the opposite decision in a time interval. For this feedback to take effect, a higher number of neighbors is required to verify the transformation request compared to the recruitment feedback. The counter is based on the majority rule (cf. feedback b).

In line 37 the robot verifies its decision by calling PELT (see Sec. II-C) using a Python package[1] with observation dictionary s_{obs} (i.e., a time series of all observations C over a time window from the last reset) whether its reset flag must be set. s_{obs} is analyzed for a change point by PELT and if found, the reset flag d_{res} is set. When the robot detects a change, it increases its reset counter r (line 25). The robot resets itself to the initial state and starts observing the environment anew (line 24). With the reset counter r we ensure that robots only influence neighbors smaller reset counters. At the end of the algorithm the robot begins broadcasting its updated robot parameters.

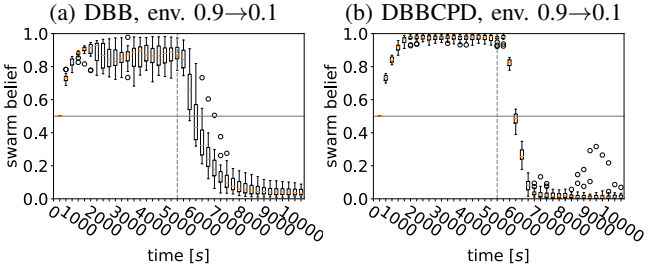
V. RESULTS

We compare this new approach with our previous contribution DBB [3]. To benchmark DBB and DBBCPD, the swarm belief of each method is monitored in different dynamic environments of varying difficulties. In the easy environment, DBB is quickly able to form a consensus in the first 750 seconds. We can observe that the median swarm belief settles at 0.99 (see Fig. 3a). This indicates the active sample limit $p_{\text{lim}} = 0.99$ which bounds the belief of each agent upward at 0.99 and downward at 0.01. Similarly in the second half, the swarm does not exceed a median swarm belief of 0.01. Considering the restriction by p_{lim}, DBB correctly classifies the environment in both halves. DBBCPD

[1] https://github.com/ritchie46/fastpelt

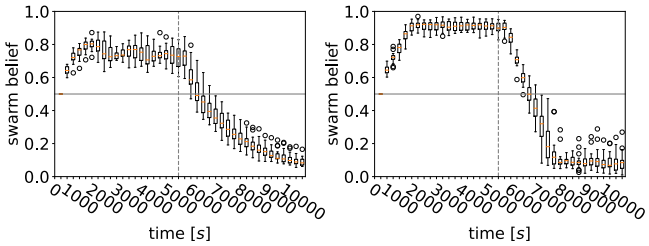


(a) DBB, env. 0.9→0.1 (b) DBBCPD, env. 0.9→0.1

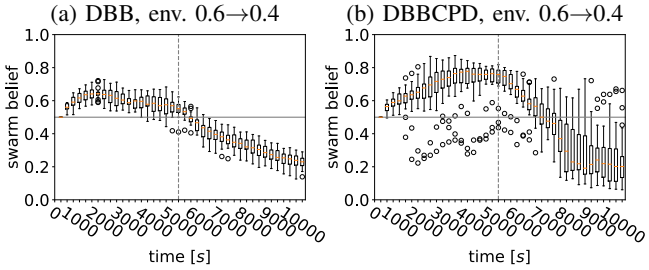


(c) DBB, env. 0.7→0.3 (d) DBBCPD, env. 0.7→0.3

Fig. 3: Comparison of DBB and DBBCPD in environments of easy (a, b) and medium (c, d) difficulty ($n = 20$ runs).



(a) DBB, env. 0.6→0.4 (b) DBBCPD, env. 0.6→0.4



(c) DBB, env. 0.55→0.45 (d) DBBCPD, env. 0.55→0.45

Fig. 4: Comparison of DBB and DBBCPD in environments of hard (a, b) and very hard (c, d) difficulty ($n = 20$ runs).

achieves a significantly better classification in both halves (see Fig. 3b, Wilcoxon rank sum test for $n = 20$ swarm beliefs each at $t = 4,990$ and $t = 10^4$, $p < 10^{-6}$). DBBCPD switches about 500 seconds faster than DBB.

Environments with medium difficulty show visible deviations. In the first half, we notice that DBB correctly classifies the environment and reaches a median swarm belief of 0.94 after 1,750 seconds (see Fig. 3c). However, the swarm oscillates between a median swarm belief of 0.83 and 0.9 until the environment changes. In the second half, variance decreases. DBB decides accurately and achieves a median swarm belief of 0.03. DBBCPD copes significantly better in the medium environment (see Fig. 3d, $p < 0.001$). We recognize its robustness, visible by low variance. In the first half, DBBCPD outperforms DBB and reaches a median swarm belief of 0.99. In the second half, DBBCPD reaches a correct median swarm belief. The decision change is faster after the environmental switch with DBBCPD. While DBB approximately needs 1,500 seconds to sink under 0.2, DBBCPD reaches this value after only 1,250 seconds.

We find clear differences between the two methods for the hard environment. Although DBB still classifies the environment correctly in both halves, the swarm only reaches a mean swarm belief of 0.8 in the first half (see Fig. 4a). Swarm belief collapses repeatedly before the change (indicating many falsely resetting agents). DBB takes more time to form the new decision after the environment change and reaches a mean swarm belief of 0.08. In comparison, DBBCPD achieves a significantly better median swarm belief of 0.93 in the first half of the hard environment ($p < 10^{-9}$), correctly classifying the predominantly white environment (see Fig. 4b). The swarm converges after 1,750 seconds. DBBCPD shows similar performance in the second half, a median swarm belief of 0.07 (not significant, $p = 0.2$). The stagnation of the swarm beliefs is caused by the limitations of the environmental difficulty. There is a constant deviation of about 0.07 from a perfect classification.

The very hard environment pushes DBB to its limits. The accuracy is not satisfactory (see Fig. 4c). The swarm merely achieves a maximum median swarm belief of 0.64 after 1750 seconds and in the end 0.2. DBBCPD reaches a significantly better median swarm belief of 0.78 in the first half of the very hard environment (see Fig. 4d, $p < 10^{-6}$). Note the belief outliers of values between 0.2 and 0.5. These may indicate that the increased change detection capabilities of DBBCPD may also be misleading sometimes. However, to test for generalization, we did not optimize hyperparameters of PELT for our specific scenario. In the second half, it improves its median swarm belief to 0.19, 3,500 seconds after the environment change (not significant, $p = 0.4$). The swarm belief stagnates at 0.78 and 0.22 in both halves. Compared to the hard environment, the limitation due to environmental difficulty has increased by 0.15.

DBBCPD is superior to DBB. DBB reaches a much lower maximum accuracy before the change. However, this low accuracy makes it initially more receptive for change. It is necessary to consider the speed in relation to a swarm's maximum achieved accuracy. DBBCPD's superiority is indicated by an at least equal speed and a higher overall swarm belief (compare Fig. 3a and Fig. 3b). DBBCPD has an advantage over DBB due to its reliable CPD in terms of accuracy. It is more robust and performs a more accurate classification of its surrounding by reducing most of the false positives.

VI. Conclusion

We presented a new Bayesian approach containing a sophisticated change detection called DBBCPD that is based on our previous work DBB [3]. Each robot executes a sophisticated, state-of-the-art change detection algorithm to analyze its local evidence. The CPD method PELT shows

good results. Its accuracy is sufficient to detect changes in hard to very hard environments. An advantage of accurate change detection is a faster and more robust consensus formation due to fewer incorrect resets before the change.

The comparison of our approach DBBCPD to our previous DBB shows a clear superiority of DBBCPD. In the hard and very hard environments DBBCPD outperforms DBB. Since DBBCPD is more accurate and faster than DBB, this is a fundamental improvement of the speed-vs-accuracy tradeoff.

The maximally achievable swarm belief decreases as expected with increasing task difficulty. There is an equilibrium between resetting and committing agents that can only be influenced by adaptability. The speed-vs-accuracy tradeoff requires balancing between conservatively resetting and committing agents and allowing for faster adaptation.

We show that DBBCPD is able to classify hard and very hard dynamic environments even without scenario specific optimization. The algorithm can be adapted to any scenario. For more results see the digital appendix.[1]

In future work, we plan to study the limits of swarm beliefs for increasing task difficulty and its connection to the speed-vs-accuracy tradeoff. This also requires to analyze the microscopic decision-making of robots that might be inhibited by contradicting local evidence. This could also help to know the actual theoretical limitations in this benchmark scenario.

Another interesting topic is to further investigate DBBCPD in more difficult and more realistic dynamic environments. Partial change or smooth transitions could help to study DBBCPD's performance and would push towards applications. The reliability of DBBCPD could be tested with Byzantine Robots (agents intentionally trying to sabotage the swarm).

The next step is to apply DBBCPD to a swarm of real Kilobots. This requires to analyze the computational costs. We use a Python package that could not be run on Kilobot but bigger robots. Here, the software for PELT needs around 82 KB of memory to process a change detection on 600 data points, far more than what a Kilobot can do. Slimmer implementations seem possible but need to be studied.

References

[1] H. Hamann, *Swarm Robotics: A Formal Approach*. Springer, 2018.

[2] G. Valentini, E. Ferrante, and M. Dorigo, "The best-of-n problem in robot swarms: Formalization, state of the art, and novel perspectives," *Frontiers in Robotics and AI*, vol. 4, p. 9, 2017.

[3] K. Pfister and H. Hamann, "Collective decision-making with bayesian robots in dynamic environments," in *2022 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 7245–7250.

[4] J. T. Ebert, M. Gauci, F. Mallmann-Trenn, and R. Nagpal, "Bayes Bots: Collective Bayesian decision-making in decentralized robot swarms," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 7186–7192.

[5] Q. Shan, A. Heck, and S. Mostaghim, "Discrete collective estimation in swarm robotics with ranked voting systems," in *2021 IEEE Symp. Series on Computational Intelligence (SSCI)*. IEEE, 2021, pp. 1–8.

[6] V. Strobel, E. Castelló Ferrer, and M. Dorigo, "Blockchain technology secures robot swarms: A comparison of consensus protocols and their resilience to Byzantine robots," *Front. in Robotics and AI*, vol. 7, 2020.

[7] G. Primiero, E. Tuci, J. Tagliabue, and E. Ferrante, "Swarm attack: A self-organized model to recover from malicious communication manipulation in a swarm of simple simulated agents," in *International Conference on Swarm Intelligence*. Springer, 2018, pp. 213–224.

[1] digital appendix: https://doi.org/10.5281/zenodo.7682100

[8] M. Raoufi, H. Hamann, and P. Romanczuk, "Speed-vs-accuracy tradeoff in collective estimation: An adaptive exploration-exploitation case," in *2021 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, 2021, pp. 47–55.

[9] G. Valentini, E. Ferrante, H. Hamann, and M. Dorigo, "Collective decision with 100 Kilobots: Speed vs accuracy in binary discrimination problems," *Journal of Autonomous Agents and Multi-Agent Systems*, vol. 30, no. 3, pp. 553–580, 2016.

[10] Q. Shan and S. Mostaghim, "Collective decision making in swarm robotics with distributed Bayesian hypothesis testing," in *International Conference on Swarm Intelligence*. Springer, 2020, pp. 55–67.

[11] P. Bartashevich and S. Mostaghim, "Multi-featured collective perception with evidence theory: tackling spatial correlations," *Swarm Intelligence*, vol. 15, pp. 83–110, 2021.

[12] J. Prasetyo, G. De Masi, and E. Ferrante, "Collective decision making in dynamic environments," *Swarm intelligence*, vol. 13, no. 3, pp. 217–243, 2019.

[13] M. S. Talamali, A. Saha, J. A. R. Marshall, and A. Reina, "When less is more: Robot swarms adapt better to changes with constrained communication," *Science Robotics*, vol. 6, no. 56, p. eabf1416, 2021.

[14] T. Aust, M. S. Talamali, M. Dorigo, H. Hamann, and A. Reina, "The hidden benefits of limited communication and slow sensing in collective monitoring of dynamic environments," in *Swarm Intelligence*, Cham, 2022, pp. 234–247.

[15] M. Divband Soorati, M. Krome, M. Mora-Mendoza, J. Ghofrani, and H. Hamann, "Plasticity in collective decision-making for robots: Creating global reference frames, detecting dynamic environments, and preventing lock-ins," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 4100–4105.

[16] S. Aminikhanghahi and D. J. Cook, "A survey of methods for time series change point detection," *Knowledge and information systems*, vol. 51, no. 2, pp. 339–367, 2017.

[17] B. Namoano, A. Starr, C. Emmanouilidis, and R. C. Cristobal, "Online change detection techniques in time series: An overview," in *2019 IEEE International Conference on Prognostics and Health Management (ICPHM)*. IEEE, 2019, pp. 1–10.

[18] C. Truong, L. Oudre, and N. Vayatis, "Selective review of offline change point detection methods," *Signal Processing*, vol. 167, p. 107299, 2020.

[19] A. B. Downey, "A novel changepoint detection algorithm," *arXiv preprint arXiv:0812.1237*, 2008. [Online]. Available: https://doi.org/10.48550/arXiv.0812.1237

[20] D. J. Cook and N. C. Krishnan, *Activity learning: discovering, recognizing, and predicting human behavior from sensor data*. John Wiley & Sons, 2015.

[21] S. Liu, M. Yamada, N. Collier, and M. Sugiyama, "Change-point detection in time-series data by relative density-ratio estimation," *Neural Networks*, vol. 43, pp. 72–83, 2013.

[22] M. Sugiyama, T. Suzuki, S. Nakajima, H. Kashima, P. von Bünau, and M. Kawanabe, "Direct importance estimation for covariate shift adaptation," *Annals of the Institute of Statistical Mathematics*, vol. 60, no. 4, pp. 699–746, 2008.

[23] T. Kanamori, S. Hido, and M. Sugiyama, "A least-squares approach to direct importance estimation," *The Journal of Machine Learning Research*, vol. 10, pp. 1391–1445, 2009.

[24] M. Yamada, T. Suzuki, T. Kanamori, H. Hachiya, and M. Sugiyama, "Relative density-ratio estimation for robust distribution comparison," *Advances in neural information processing systems*, vol. 24, 2011.

[25] E. S. Page, "Continuous inspection schemes," *Biometrika*, vol. 41, no. 1/2, pp. 100–115, 1954.

[26] K. Yamanishi and J.-i. Takeuchi, "A unifying framework for detecting outliers and change points from non-stationary time series data," in *ACM SIGKDD on Knowl. discov. & data mining*, 2002, pp. 676–681.

[27] Y. Saatçi, R. D. Turner, and C. E. Rasmussen, "Gaussian process change point models," in *ICML*, 2010.

[28] R. P. Adams and D. J. MacKay, "Bayesian online changepoint detection," *arXiv preprint arXiv:0710.3742*, 2007. [Online]. Available: https://doi.org/10.48550/arXiv.0710.3742

[29] R. Killick, P. Fearnhead, and I. A. Eckley, "Optimal detection of changepoints with a linear computational cost," *Journal of the American Statistical Association*, vol. 107, no. 500, pp. 1590–1598, 2012.

[30] B. Jackson *et al.*, "An algorithm for optimal partitioning of data on an interval," *IEEE Signal Proc. Letters*, vol. 12, pp. 105–108, 2005.

[31] J. Ebert and R. Barnes, "Kilosim," Mar. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.3406865