Discrete collective estimation in swarm robotics with distributed Bayesian belief sharing

Qihao Shan[1] · Sanaz Mostaghim[1]

Abstract

Multi-option collective decision-making is a challenging task in the context of swarm intelligence. In this paper, we extend the problem of collective perception from simple binary decision-making of choosing the color in majority to estimating the most likely fill ratio from a series of discrete fill ratio hypotheses. We have applied direct comparison (DC) and direct modulation of voter-based decisions (DMVD) to this scenario to observe their performances in a discrete collective estimation problem. We have also compared their performances against an Individual Exploration baseline. Additionally, we propose a novel collective decision-making strategy called distributed Bayesian belief sharing (DBBS) and apply it to the above discrete collective estimation problem. In the experiments, we explore the performances of considered collective decision-making algorithms in various parameter settings to determine the trade-off among accuracy, speed, message transfer and reliability in the decision-making process. Our results show that both DC and DMVD outperform the Individual Exploration baseline, but both algorithms exhibit different trade-offs with respect to accuracy and decision speed. On the other hand, DBBS exceeds the performances of all other considered algorithms in all four metrics, at the cost of higher communication complexity.

1 Introduction

Collective decision-making is a field in swarm intelligence that has long been studied. Researchers have observed that naturally occurring swarm intelligent systems, such as insect swarms and bird flocks, have no centralized control mechanisms, and individual agents make decisions purely via local interactions with their surrounding environments

---

✉ Qihao Shan
  qihao.shan@ovgu.de

  Sanaz Mostaghim
  sanaz.mostaghim@ovgu.de

[1] Faculty of Computer Science, Otto-von-Guericke University Magdeburg, Universitätsplatz 2, 39106 Magdeburg, Germany

or their neighbors (Camazine et al. 2001). Researchers in this field aim to understand the mechanisms that enable decision-making in naturally occurring swarm intelligent systems, as well as construct decision-making strategies that enable groups of artificial intelligent agents to collectively make decisions. By constructing decision-making mechanisms using local interactions among a group of intelligent agents, the decision-making process of the whole swarm can be made robust, flexible and scalable (Şahin 2005).

Best-of-n problems focus on enabling consensus-forming using various collective decision-making strategies when a group of agents are choosing from a set of options (Valentini et al. 2017). These problems can have many different scenarios, such as route picking, site selection or collective perception. In this paper, we focus on the collective perception scenario, which is a best-of-n problem with discrete options with asymmetric qualities. The collective perception scenario was first introduced in Valentini et al. (2016a). The setting of a typical collective perception scenario is as follows. There is an arena with black and white tiles. The proportion of black tiles in the arena is referred to as the fill ratio. A number of mobile robots roam the arena, and their goal is to collectively agree on which color is in the majority, hence whether the fill ratio is above or below 0.5. The robots are assumed to have limited communication and sensory abilities.

Various collective decision-making strategies have been used to perform collective perception in the past. Valentini et al. (2014) proposed direct modulation of voter-based decisions (DMVD). They also proposed direct modulation of majority-based decisions (DMMD) in Valentini et al. (2015) and further analyzed it in Valentini et al. (2016b). In addition, direct comparison (DC) is proposed as a benchmark algorithm (Valentini et al. 2016a). Strobel et al. (2018) have investigated the performances of these collective decision-making strategies at different ratios of black and white tiles. Bartashevich and Mostaghim (2019) have investigated collective perception scenarios with different patterns of black and white tiles and their impact on the performances of collective decision-making strategies. Ebert et al. (2020) have employed a Bayesian statistics-based approach to perform collective perception on Kilobots. So far, most investigations of collective perception scenarios have been limited to the task of determining the color in majority, thus limiting the number of options for agents to only 2. As Valentini et al. (2017) have noted, this is a common constraint across the best-of-n problems and there are only a few works into multi-option collective decision-making scenarios. Specifically in collective perception problems, the robots are usually restricted to binary choices of the color in majority.

Many option to continuous consensus problems have been studied extensively for networked sensors and agents (Olfati-Saber et al. 2006; Wang and Xiao 2010). These algorithms, however, cannot be directly applied to a swarm robotics setting, as they usually require high complexity in robot design and frequent communications. In a swarm robotics setting, extending best-of-n problems to multiple options can have different implications for different application scenarios. In route picking or site selection, the agents can choose from a larger set of possible options. For example, Garnier et al. (2013) and Scheidler et al. (2016) have investigated selection of the shortest path by a robot swarm among multiple choices using collective reinforcement of the optimal route. For collective perception, extending the number of options means the robots would choose from a series of discretized fill ratios and form a more accurate estimation of the fill ratio in the arena. Similar collective estimation tasks have been attempted in Strobel and Dorigo (2018), who used a blockchain-based method to create a shared knowledge, which is a piece of information accessed by all members of the swarm, for all the robots, who would pool their estimations of the fill ratio. The mean of all estimations would be the final estimation by the whole swarm. Shan and Mostaghim (2020) proposed an algorithm using distributed

Bayesian hypothesis testing and centralized opinion fusion to determine the most likely fill ratio hypothesis. In their algorithm, the robots form a communication network to pass their individual beliefs to a leader, who computes the combined opinion of the whole swarm.

In this paper, we employ collective decision-making strategies with high scalability and low complexity to solve a discrete collective estimation problem. This paper is a significant extension to Shan and Mostaghim (2020) in the following ways. Firstly, on the top of a similar Bayesian hypothesis testing technique used to compute option qualities, we propose a new decentralized decision-making strategy, distributed Bayesian belief sharing, that avoids the single point of failure introduced by the leader in Shan and Mostaghim (2020). Secondly, we thoroughly investigate the discrete collective estimation problem and test the performances of other state-of-the-art collective decision-making strategies, such as DMVD and DC, in a multi-option collective decision-making scenario. After that, we compare the performances of considered algorithms in discrete collective estimation scenarios using a multi-objective optimization framework and investigate the various trade-offs in parameter selection for different decision-making strategies in the considered scenario. Finally, we also examine the performance of our proposed algorithm in scenarios with sparse communications.

The structure of this paper is as follows. Section 2 includes the problem statement and an analysis of related works in similar problems. In Sect. 3, we present our method to compute the option quality used in all considered collective decision-making strategies. We then present our considered benchmark algorithms in this paper, as well as our proposed algorithm DBBS. After that in Sect. 4, we provide our experiments and analyze the results. In Sect. 5, we will discuss the results and compare them with other related studies. Finally, Sect. 6 contains the conclusion.

2 Problem statement and related works

We consider the following discrete collective estimation scenario, illustrated in Fig. 1. The name of the scenario investigated is based on the name used by Strobel and Dorigo (2018). This scenario is an extension of the collective perception scenario in Valentini et al. (2016a), which is well investigated in the field of collective decision-making. In the scenario investigated here, there is an arena as shown in Fig. 1 with a number of tiles that can be either black or white. The black tiles are of a particular fill ratio within the arena. N mobile robots roam the arena, shown in red in Fig. 1, and their goal is to collectively determine the most likely fill ratio out of a number of discrete hypotheses. The robots have limited capabilities in sensing, communication and computation. They can only communicate with other robots within a communication radius, are only able to sense the color of the tiles directly beneath them and have only simple reactive behavior.

Direct comparison of option qualities (DC) was proposed by Valentini et al. (2016a) and remains a popular benchmark algorithm for various collective decision-making scenarios, such as collective perception (Strobel et al. 2018; Bartashevich and Mostaghim 2019; Shan and Mostaghim 2020) and site selection (Talamali et al. 2019). When applied to collective decision-making problems where the option qualities must be measured individually, such as site selection, DC faces a problem that an agent who receives a preferred option and its quality from its neighbor cannot verify the accuracy of the reported option quality. Therefore, erroneous measurements can be propagated across the agents, thus reducing the accuracy of the consensus. As a result, cross-inhibition, which is proposed by Reina et al.

Fig. 1 An example of the arena which is a 2 m × 2 m square, covered in 400 square tiles shown in black and white. The red dots illustrate 20 mobile robots (Color figure online)
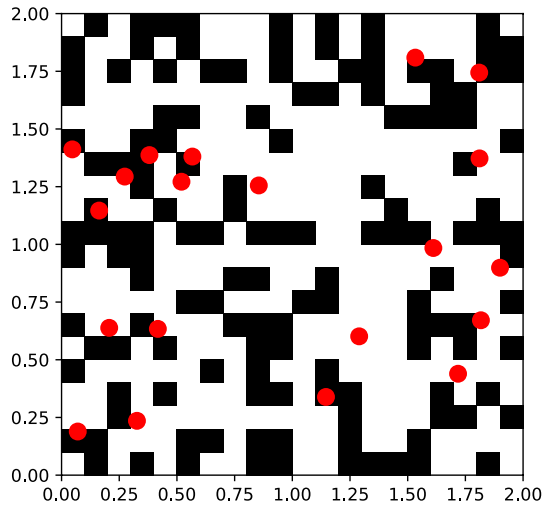
(2015), is used by Talamali et al. (2019) to perform site selection. However, in our discrete collective estimation scenario, the qualities of all potential options can be measured simultaneously, as explained in Section 3.1. Therefore, we follow the literature on collective perception and still use DC as a baseline algorithm. DC decision-making strategy in collective perception works as follows. Low-level controllers direct the robots to perform random walk in the arena. Each agent alternates between two decision-making states, exploration and dissemination, with an identically distributed but stochastically varying duration. During exploration, the agents sample the arena and compute the quality of the current own option, which in collective perception is the likelihood that their chosen color is in the majority. During dissemination, the agents broadcast the current option and quality to their neighbors. At the end of a dissemination period, an agent switches to an option out of the options of its neighbors and itself which has the highest quality.

Direct modulation of voter-based decisions (DMVD) was first proposed by Valentini et al. (2014). It has also been used to perform site selection (Valentini et al. 2016b) and collective perception (Valentini et al. 2016a; Strobel et al. 2018; Bartashevich and Mostaghim 2019; Shan and Mostaghim 2020). DMVD works similar to DC when applied to collective perception. Agents also alternate between exploration and dissemination states. However, the length of agents' dissemination states is proportional to the option qualities. In addition, at the end of a dissemination period, an agent switches to a random option out of the options of its neighbors and itself.

In this paper, we use a different method from those stated above. Our method is based on Bayesian statistics. Bayesian statistics has long been used in sensor fusion techniques in sensor networks. Hoballah and Varshney (1989) and Varshney and Al-Hakeem (1991) have designed various decentralized detection algorithms based on Bayesian hypothesis testing in sensor networks. Alanyali et al. (2004) explored how to make multiple connected noisy sensors reach consensus using message passing based on belief propagation. Similar fusion of opinions has been attempted in the field of collective decision-making in the form of opinion pooling (Lee et al. 2018a, b; Crosscombe et al. 2019), where small groups of agents in a swarm would combine their opinions iteratively until consensus is reached for the whole swarm. However, implementations of opinion pooling remain mostly

in non-physics-based simulations. When applied to the discrete collective estimation scenario, opinion pooling requires bidirectional communications between the robots as well as more complex control of the robots during the pooling process so that every agent taking part in the pooling can have access to the results. Therefore, we do not take this approach in this paper. Apart from that, the following two approaches have been proposed for collective perception based on Bayesian statistics. Ebert et al. (2020) employed a Bayesian statistics-based collective perception algorithm. In order to obtain the color in majority in their algorithm, each robot is programmed to compute the probability of the fill ratio to be larger than 0.5. They employed bio-inspired positive feedback to help the decision-making. Shan and Mostaghim (2020) have used Bayesian hypothesis testing to compute individual robots' belief on the fill ratio. We use a similar method in this paper to compute the option qualities.

3 Methodology

In this section, we introduce the collective decision-making strategies being investigated in this paper in details. We first introduce the methodology to compute option qualities and the low-level control mechanisms, both of which are used for all considered decision-making strategies. Then, we will show the inner workings of the considered strategies.

3.1 Computing the option qualities

For the discrete collective estimation scenario, we can use Bayesian statistics to enable a single robot to easily compute the likelihoods of all hypotheses, based on its past observations of the environment. We use the likelihood of a particular hypothesis as its option quality. The technique used in Shan and Mostaghim (2020) to compute individual belief of fill ratio is used here to compute the option qualities for all considered decision-making strategies.

From the perspective of an individual robot, the environment is seen as a discrete random variable V with two possible values *Black* and *White*. $P(V = Black) = P_B$ is the probability that a random place in the arena is black, and thus, it is equal to the proportion of area in the arena that is covered by black tiles. At a given point of time, a robot has made S observations of the environment, and it could then compute the likelihood of a given hypothesis h of the value of P_B using its past observations, expressed as $P(P_B = h|ob_1...ob_S)$. We can then use Bayes' rule

$$P(P_B = h|ob_1...ob_S) = \frac{P(ob_1...ob_S|P_B = h)P(P_B = h)}{P(ob_1...ob_S)}. \tag{1}$$

Here, $P(P_B = h)$ is the prior and $P(ob_1...ob_S)$ is the marginal likelihood, both of which we assume to be the same for all hypotheses. We can then apply the chain rule to $P(ob_1...ob_S|P_B = h)$

$$= P(ob_1|P_B = h)P(ob_2|P_B = h, ob_1)...P(ob_S|P_B = h, ob_1, ..., ob_{S-1}). \tag{2}$$

Assuming the observations are all independent of each other, we can remove the dependencies on previous observations, and thus, we have:

$$= P(ob_1|P_B = h)P(ob_2|P_B = h)...P(ob_S|P_B = h). \tag{3}$$

In practice, we express the quality of option h as the normalized likelihood of the hypothesis given previous observations. We arrange all the available hypotheses of black tile fill ratios as matrix H as follows. The first column is the proportion of black tiles (P_B), and the second column is the proportion of white tiles ($1 - P_B$).

$$H = \begin{bmatrix} 0.05 & 0.95 \\ 0.15 & 0.85 \\ ... & \\ 0.95 & 0.05 \end{bmatrix} \tag{4}$$

An observation can be either $\underline{ob} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ for black or $\underline{ob} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ for white tiles. Therefore, the vector including the quality of all options for a robot can be calculated:

$$\underline{\rho} = \Pi_{s=1..S}H \cdot \underline{ob}_s. \tag{5}$$

In practice, the computation of qualities can be iteratively done by a robot, at very low cost in terms of computational power and memory. The robot starts with a prior belief that all options are equally likely. At a fixed sampling interval, the robot collects an observation sample of the arena floor and multiplies its old belief with a new set of probabilities.

$$\underline{\rho}_{n,0} = \begin{bmatrix} 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \end{bmatrix}^T \tag{6}$$

$$\underline{\rho}_{n,s} = \underline{\rho}_{n,s-1} \circ (H \cdot \underline{ob}_s). \tag{7}$$

In order to maintain the independence assumption made in Eq. (3), we made the following design choices during implementation of our considered algorithms. First, we use a sampling interval to be large enough to avoid successive observation samples being collected on one tile, which can cause a high level of correlation between them. In addition, in all considered algorithms, the robot can collect an observation sample only when it is moving forward and not when it is turning. This is also to avoid multiple samples being collected on one tile.

3.2 Low-level control mechanism

In all considered decision-making strategies, a low-level controller as shown in Fig. 2 directs the robots to perform a random walk in the arena similar to in Valentini et al. (2016a). The controller has the structure of a finite state machine with two states. In State A, the robot will move forward, while in State B, the robot turns in place toward a random direction. The lengths of the two states are randomly distributed. The durations of State A are exponentially distributed with a mean of 40 s, while the durations of State B are uniformly distributed between 0 s and 4.5 s. The robots are assumed to be able to detect obstacles 0.1 m located in front them. To avoid collisions, during State A, if an obstacle is detected, the robot enters State B. At the end of State B, if an obstacle is detected, the robot repeats State B with a reset timer.

It has been observed in the related literature which used Bayesian statistics in collective perception (Ebert et al. 2020; Shan and Mostaghim 2020) that collecting correlated
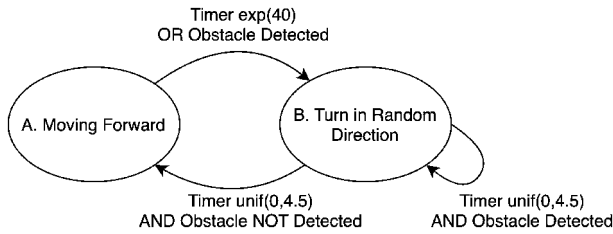
Fig. 2 Low-level control mechanism used to perform random walk

observations is detrimental to accurate decision-making. When the robot is turning in State B, it remains in the same place, and thus, any observations it collects are of the same color. Therefore, for all considered decision-making strategies, we restrict the observation collection to only when the robots are moving forward in State A.

3.3 Benchmark strategies: Individual Exploration, DC and DMVD

In this section, we introduce the benchmark decision-making strategies used in this paper. Individual Exploration is a simple baseline strategy to determine whether the adoption of a particular collective approach is beneficial. Both DC and DMVD are collective decision-making strategies used in past related works, but mostly for binary decision-making problems.

3.3.1 Individual Exploration

Individual Exploration algorithm 1 is used as a baseline. In this algorithm, the robots perform a random walk in the arena. At every control loop, the robot with index n samples the arena floor for an observation (line 3) and computes its own belief of the fill ratio hypotheses $\underline{\rho}_n$. Its decision d_n is always the option that has the highest likelihood by its own estimation. The algorithm terminates when the decisions for all robots converge to a single option (i.e., $d_1 = d_2 = \cdots = d_n$). This algorithm does not employ any communication and is used as a baseline in this paper. A collective decision-making algorithm needs to perform significantly better than the Individual Exploration baseline to justify its extra infrastructure as well as potential security and defect risk introduced by communication.

Algorithm 1: Individual Exploration Baseline

   Input   : Initialized belief of robot n: $\underline{\rho}_n$
   Output: Converged decision: d_n
1 **while** *Decisions in swarm have not converged* **do**
2   **if** *Robot n is moving forward in motion State A* **then**
3 $\underline{ob}$ = CollectObservation
4 $\underline{\rho}_n = Normalize(\underline{\rho}_n \circ (H \cdot \underline{ob}))$
5 $d_n = MaxIndex(\underline{\rho}_n)$

3.3.2 Direct comparison (DC) and direct modulation of voter-based decisions (DMVD)

Algorithms 2 and 3 show the direct comparison and direct modulation of voter-based decisions strategies. We have modified the original algorithms from Valentini et al. (2016a) to a multi-option scenario. We have used Bayesian hypothesis testing as shown in Section 3.1 to compute the option qualities (lines 9–10 in Algorithm 2 and Algorithm 3).

Algorithm 2: Direct Comparison of Option Qualities (DC)

   Input: Initialized belief of robot n: $\underline{\rho}_n$, Initialized decision: d_n
   Output: Converged decisions: d_n
1 Set σ, τ
2 Initialize d_n with random valid values
3 $state_n = 0$, $timer_n = 0$
4 **while** *Decisions in swarm have not converged* **do**
5 $timer_n = timer_n - 1$
6   **if** $state_n = 0$ **then**
7   #Exploration State
8   **if** *Robot n is moving forward in motion State A* **then**
9 $\underline{ob}$ = CollectObservation
10 $\underline{\rho}_n = Normalize(\underline{\rho}_n \circ (H \cdot \underline{ob}))$
11   **if** $timer_n < 0$ **then**
12 $state_n = 1$
13 $timer_n = Sample(exp(\sigma))$
14   **else**
15   #Dissemination State
16 (d_m, ρ_{m,d_m}) = CollectNeighborOpinion
17   Broadcast(d_n, ρ_{n,d_n})
18   **if** $timer_n < 0$ **then**
19   **if** $max(\rho_{m,d_m}) > \rho_{n,d_n}$ **then**
20 $d_n = d_{m'}$, where m' is the index of the robot with $argmax(\rho_{m,d_m})$
21 $d_n = RandomChoice([(d_n + 1), (d_n - 1)], \tau)$
22 $state_n = 0$
23 $timer_n = Sample(exp(\sigma))$

In a collective decision-making problem with only two options, if the agents have not converged to a single option, all options would have at least one agent choosing it. Therefore, there is no need to maintain a diversity of opinions among the agents. However, when applied to a scenario with more than two options, the swarm needs to keep a diversity of opinions among the robots during the decision-making process to test all of the options and reach an accurate consensus. In the original decision-making mechanism of DC and DMVD, a robot can only

switch to an option, if there is a neighbor holding that specific option. Therefore, it is a likely situation that all robots holding a particular option have switched to other options, causing some options to be eliminated prematurely. This can result in the reduction of the diversity in robots' opinions with no way to recover. Here, we introduce a new mechanism to let the robot randomly switch from d_n to a new option that is next to its old option (i.e., $d_n + 1$ or $d_n - 1$) with a probability of τ at the end of dissemination periods (Function RandomChoice, line 21 in Algorithm 2, line 20 in Algorithm 3). τ is a design parameter and is set by the user.

Another design parameter σ is the average length of exploration and dissemination periods. As observed in Brambilla (2015), varying the average length of exploration and dissemination periods has a significant influence on decision-making speed and accuracy for both DC and DMVD. During the exploration states (lines 7–13), the agents collect observations of the color of the arena tile beneath them and then compute their own belief of the arena composition $\underline{\rho}_n$. During the dissemination states (lines 15–22), the agents broadcast their opinions to their neighbors and record all their neighbors' broadcasted opinions. The rate of the message broadcasts for both algorithms during dissemination is once per control loop. In practice, robots additionally need to broadcast a unique identification number to avoid repetitively logging the same opinions, as implemented in Valentini et al. (2016b).

Algorithm 3: Direct Modulation of Voter-based Decisions (DMVD)

Input: Initialized belief of robot n: $\underline{\rho}_n$, Initialized decision: d_n
Output: Converged decisions: d_n

1 Set σ, τ
2 Initialize d_n with random valid values
3 $state_n = 0$, $timer_n = 0$
4 **while** *Decisions in swarm have not converged* **do**
5      $timer_n = timer_n - 1$
6   **if** $state_n = 0$ **then**
7   #Exploration State
8   **if** *Robot n is moving forward in motion State A* **then**
9              $\underline{ob} = $ CollectObservation
10             $\underline{\rho}_n = Normalize(\underline{\rho}_n \circ (H \cdot \underline{ob}))$
11   **if** $timer_n < 0$ **then**
12             $state_n = 1$
13             $timer_n = $ Sample$(exp(\sigma\rho_{n,d_n}))$
14   **else**
15   #Dissemination State
16         $d_m = $ CollectNeighborOpinion;
17   Broadcast(d_n)
18   **if** $timer_n < 0$ **then**
19             $d_n = d_{m'}$, where m' is the index of a random robot whose opinion is collected in the current dissemination interval
20             $d_n = $ RandomChoice$([(d_n + 1), (d_n - 1)], \tau)$
21             $state_n = 0$
22             $timer_n = $ Sample$(exp(\sigma))$

3.4 Distributed Bayesian belief sharing

In the following, we introduce our proposed collective decision-making strategy called distributed Bayesian belief sharing (DBBS), as shown in Algorithm 4. In this approach, a robot with

index n keeps two sets of beliefs, $\underline{\rho}_n$ and $\underline{\rho}'_n$. $\underline{\rho}_n$ denotes the belief computed from the robot's own observations, and $\underline{\rho}'_n$ denotes the combined belief received from the other robots.

At every control loop, the robot collects a sample of the color of the arena floor and modifies $\underline{\rho}_n$ (lines 4–6). The vector $\underline{\xi}$ is used to denote the messages passed among the robots. The robot also receives a neighbor's message $\underline{\xi}_m$ from a random neighbor if there is one (line 6). $\underline{\xi}_m$ encodes the belief of robot m. The communication paradigm here is different from in DC and DMVD in that an agent is restricted to receiving only one message in a control loop. If multiple other agents are present in the communication radius, a random agent's belief among them would be received.

If $\underline{\xi}_m$ is received, the robot uses it to update its record of its neighbors' belief $\underline{\rho}'_n$, which is formulated as follows (line 8):

$$\underline{\rho}'_n = Normalize(\underline{\rho}'^{\lambda}_n \circ \underline{\xi}_m)$$

$\underline{\rho}'_n$ is updated with a weighted product of itself and $\underline{\xi}_m$ from a neighbor. λ is the decay coefficient of past neighbors' beliefs. It is the weight of $\underline{\rho}'_n$ itself during the update and is also a parameter set by the user. λ is applied so that the robot forgets older $\underline{\xi}$ messages from neighbors by counting them progressively less when new $\underline{\xi}$ messages are available.

The robot then computes its own message $\underline{\xi}_n$ (line 9) which is a weighted element-wise product of $\underline{\rho}_n$ and $\underline{\rho}'_n$, with the weight of $\underline{\rho}'_n$ being μ, shown as follows:

$$\underline{\xi}_n = \underline{\rho}_n \circ \underline{\rho}'^{\mu}_n.$$

μ is the decay coefficient of current neighbors' beliefs in communication. Finally, the robot broadcasts $\underline{\xi}_n$ to its neighbors for the rest of the control loop (line 10).

A robot's decision is the corresponding option of the highest combined quality computed as follows:

$$d_n = MaxIndex(\underline{\rho}_n \circ \underline{\rho}'_n)$$

The two parameters λ and μ control different mechanisms in the decision-making process. λ controls the speed that old beliefs received from neighbors decay. Higher λ means that the robots do not easily forget old beliefs sent to it by its neighbors and vice versa, while μ is the rate at which the current neighbors' belief decays during communication, and it controls the degree of positive feedback among the robots in the decision-making process. The higher the μ is, the bigger the effect of positive feedback among the robots is and vice versa. Both parameters have a range between 0 and 1.

Increasing either λ or μ strengthens the impact of neighboring robots' opinions relative to a robot's own opinion during decision-making and therefore increases the decision-making speed.

Positive feedback has been used before in collective decision-making. In DC and DMVD, as well as the Bayesian algorithm proposed by Ebert et al. (2020), positive feedback is used, but only on the collective level, which means that the decision made by a particular robot induces other robots to make the same decision. However, we also employ positive feedback on an individual level, which means that a robot's belief would strengthen itself as it would be passed back from its neighbors in a decayed form. A robot's belief does not only influence its immediate neighbors, but also is passed along in the swarm. Such design causes the beliefs of all agents in the swarm to be more closely and more

directly linked, therefore making the decision-making process faster without needing to sacrifice accuracy.

Algorithm 4: Distributed Bayesian Belief Sharing (DBBS)

Input: Initialized belief of robot n: $\underline{\rho}_n$, Initialized record of neighbors' beliefs : $\underline{\rho}'_n$
Output: Converged decisions: d_n

1 Set λ, μ
2 **while** *Decisions in swarm have not converged* **do**
3   **if** *Robot n is moving forward in motion State A* **then**
4          $\underline{ob}$ = CollectObservation
5          $\underline{\rho}_n = Normalize(\underline{\rho}_n \circ (H \cdot \underline{ob}))$
6      $\underline{\xi}_m$ = CollectNeighborOpinion
7   **if** $\underline{\xi}_m$ *collected* **then**
8          $\underline{\rho}'_n = Normalize(\underline{\rho}'^{\lambda}_n \circ \underline{\xi}_m)$
9      $\underline{\xi}_n = \underline{\rho}_n \circ \underline{\rho}'^{\mu}_n$
10   Broadcast($\underline{\xi}_n$)
11     $d_n = MaxIndex(\underline{\rho}_n \circ \underline{\rho}'_n)$

---

4 Experiments

In this section, we describe our experiments and analyze the results. The settings of our experiments are largely the same as (Valentini et al. 2016a; Strobel et al. 2018; Bartashevich and Mostaghim 2019; Shan and Mostaghim 2020)[1]. The arena is $2\,\text{m} \times 2\,\text{m}$ with 400 tiles as shown in Fig. 1. We use 20 simulated robots with the specification of e-pucks (Mondada et al. 2009) to perform the designated task. The linear speed of the robots is set to 0.16 m/s, and the rotational speed is set to 0.75 rad/s. As for the maximum communication radius, we follow the settings used in the previous literature and set it to 0.5 m. The simulation is done by mapping out all agents' positions using a NumPy array in Python. We then iteratively update the system in time steps of 0.01 s and simulate the movement, sensing and communication. Both sensing and communication are assumed to be accurate.

In Shan and Mostaghim (2020), it is concluded that a sampling interval of 1 second is suitable for Bayesian hypothesis testing in this environment. Additionally, we aim to keep the number of observations and exchanged messages among the robots at similar levels across the considered algorithms. For DMVD and DC, an observation is collected on average once every two control loops. The rate of message transmission is also at most once every two control loops. We therefore set the length of control loops to be 1 s. For Individual Exploration baseline and DBBS, an observation is made every control loop and a message transmission happens every control loop as well. Therefore, we set the length of control loops for these two algorithms to be 2 s in order to keep a fair comparison.

To measure the performances of considered algorithms in a particular environment setting, we measure the mean absolute error, mean consensus time, mean number of message transfers and the failure rate. The related literature on collective perception (Valentini et al. 2016a; Strobel et al. 2018; Bartashevich and Mostaghim 2019; Shan and Mostaghim

---

[1] Our codes are available at: https://github.com/shanqihao/CDM_Discrete_Collective_Estimation.

2020) used mostly exit probability and consensus time. However, this measurement does not differentiate between reaching a wrong consensus and not being able to reach a consensus at all. Therefore, we use the mean absolute error between the obtained consensus and the actual fill ratio to measure the accuracy of considered algorithms. Here, consensus is reached when all robots are choosing the same option. For some experimental runs, the robots are not able to come to a consensus after a long time. We therefore use failure rate to measure the probability of a considered algorithm to fail to reach a consensus in the given time limit of 1200 seconds. The time limit of 1200 seconds is selected to be well above the average consensus time for considered algorithms to reduce the probability of premature rejection of an experimental run. Since the investigated algorithms have different communication paradigms, we additionally measure the mean number of message transfers during the decision-making process. Thus, we can take an integrated look at the investigated algorithms and study the trade-off between the performance and the required communication overhead.

4.1 Random environmental feature distribution

We first test the considered algorithms in environments with random distribution of black and white tiles, at different fill ratios ranging from 0.05 to 0.45, with 50 experimental runs in each environmental and design configuration. All results obtained are averaged over all five fill ratio scenarios.

4.1.1 Numerical results

It has been previously observed in the literature (Brambilla 2015; Ebert et al. 2020; Shan and Mostaghim 2020) that there is a trade-off between the speed and the accuracy in the parameter selection of the collective perception algorithms. Therefore, we measure the performance of the considered algorithms at various parameter settings to investigate this trade-off. The performances of DC at various σ and τ settings are shown in Table 1. It can be observed that as either σ or τ is increasing, the error generally decreases, while the consensus time and failure rate increase. Also, the design configurations with no diversity enforcement, i.e., $\tau = 0$, produce the highest errors. Moreover, the decrease in error is steeper and the increase in consensus time is milder in the increasing τ direction. Therefore, when trying to increase the accuracy by sacrificing consensus time, it is better to have a high τ value than a high σ value. This is demonstrated in the following data points. At $\sigma = 1$ s and $\tau = 0.05$, the mean error is 0.196 and the mean consensus time is 162.0 s. However, to reach a similar consensus time of $166s$ by increasing σ to 8 while keeping τ at 0, we obtain a higher error of 0.534. This is because the decision-making mechanism of DC can quickly lead the decisions in a neighborhood to converge to the option with the highest quality, and thus, it is more beneficial for achieving a more accurate decision-making to maintain a high diversity of opinions. A high τ value means that the robots readily and randomly modify their opinions. If a modification is beneficial, it will soon spread across neighboring robots, and if a modification is not beneficial, the robot will quickly switch to a better decision in the next dissemination session.

In addition, the failure rate is quite low and around 0 across all parameter settings except at high σ settings. This shows that DC is very reliable in providing a consensus even in a

multi-option collective decision-making problem. The mean number of message transfers is largely proportional to the mean consensus time. This is because DC has a constant rate of message transfer.

The performance of DMVD at various σ and τ settings is shown in Table 2. Here, the impacts of σ and τ are similar to in DC. The configurations with $\tau = 0$ also produce more error than other setting with larger τ values. However, for DMVD, increasing σ has a larger impact on reducing the error and increasing the consensus time than increasing τ. In addition, increasing either σ or τ introduces a significant failure rate. This is shown via the following data points. At $\sigma = 1$ s and $\tau = 0.05$, the mean error is 1.197 and the mean consensus time is 678.1 s. However, to reach a similar mean error by increasing σ to 8 s while holding τ at 0, we obtain a mean consensus time of 428.6 s, which is far smaller but at a similar mean error of 1.010. This is because DMVD uses sampling to select options for individual robots and therefore is able to maintain a high level of diversity in the opinions of the robots during the decision-making process. If more diversity is added via a high τ setting, the robot can have difficulty coming to a consensus, as shown in the high failure rate when τ is high. The mean number of message transfers for DMVD is not strictly proportional to the consensus time. This is because DMVD changes the lengths of the agents' dissemination states depending on the qualities of their decisions. Thus, agents often do not keep broadcasting their decisions for the full duration set by the parameter σ.

The performance of DBBS at various λ and μ settings is shown in Table 3. Similar to DC and DMVD, DBBS also faces a trade-off between the speed and the accuracy. With either λ or μ increasing, the error increases, while the consensus time decreases. The increase in error is very significant when μ increases from 0.6 to 1.0 but not significant before that. Also, the decrease in consensus time is significant when μ increases from 0 to 0.4, but is also not significant beyond that. Therefore, the selection of μ and hence the degree of positive feedback in the decision-making process has an optimum at around 0.4 and 0.6 that can achieve both fast and accurate decision-making relative to other μ values. In contrast, increasing λ has a steadier impact of the performance and the speed vs accuracy trade-off is more prevalent here. On the other hand, the mean number of message transfers is also roughly proportional to the consensus time. In addition, DBBS has not had a failure in decision-making during our experiments in this scenario, which shows its robustness compared to the other considered algorithms.

4.1.2 Multi-objective analysis

The above numerical results are depicted in Figs. 3 and 4 so that we can investigate the trade-offs between the considered metrics for the different algorithms. Figure 3 depicts the trade-off between consensus time and mean absolute error. The performance of the Individual Exploration baseline is also shown in the figure (+ Marker). Individual Exploration can ensure zero error in the final consensus when the decision-making process is successful. This is because in Individual Exploration, robots do not communicate with each other. Therefore, for a consensus to be formed, all robots must pick that option based on their own observations. The probability for all robots to converge to a wrong option is almost zero. However, due to this design, Individual Exploration has a high mean consensus time of 651 s and a high failure rate of 0.332, as there is no way to encourage the formation of a consensus among the robots.

Table 1 Mean absolute error, mean consensus time, mean message transfer and failure rate of DC at different σ and τ settings; vertical-σ mean lengths of exploration/dissemination states, horizontal-τ probability of mutating the chosen option

Mean Error/0.1							Mean Consensus Time/s					
τ	0.00	0.01	0.02	0.03	0.04	0.05	0.00	0.01	0.02	0.03	0.04	0.05
σ/s 1	0.736	0.588	0.392	0.380	0.260	0.196	71.8	84.4	101.5	116.3	134.7	162.0
2	0.804	0.472	0.452	0.332	0.244	0.180	77.4	98.8	112.0	135.2	150.3	168.7
3	0.712	0.420	0.384	0.364	0.228	0.196	99.7	120.8	130.4	138.3	162.8	186.5
4	0.588	0.460	0.360	0.292	0.184	0.152	116.4	131.1	138.1	151.4	174.8	198.0
5	0.610	0.296	0.348	0.240	0.180	0.144	129.2	142.3	157.0	183.1	184.7	196.7
6	0.470	0.348	0.320	0.224	0.196	0.140	150.0	152.4	163.1	177.1	188.3	203.8
7	0.512	0.308	0.356	0.244	0.220	0.148	147.3	169.1	170.9	194.6	196.6	225.7
8	0.534	0.316	0.280	0.188	0.192	0.116	166.4	171.8	182.2	203.7	210.3	243.5
9	0.456	0.336	0.324	0.236	0.168	0.120	165.6	181.1	193.5	211.8	233.7	239.2
10	0.400	0.364	0.272	0.156	0.200	0.156	179.0	189.4	208.3	230.0	224.3	244.5
15	0.351	0.284	0.244	0.196	0.116	0.092	218.6	249.2	248.5	269.9	278.7	307.1
20	0.294	0.229	0.217	0.192	0.100	0.084	248.0	267.7	291.7	295.7	334.1	357.0
25	0.281	0.252	0.172	0.148	0.160	0.100	269.4	316.8	342.3	348.5	364.4	386.1
30	0.305	0.208	0.180	0.136	0.120	0.104	326.0	348.3	366.5	393.6	398.1	425.7
35	0.325	0.209	0.192	0.132	0.104	0.069	359.3	383.3	391.2	418.9	443.7	468.6
40	0.292	0.241	0.150	0.137	0.116	0.073	400.9	405.9	426.6	452.7	458.4	506.9
45	0.309	0.194	0.153	0.104	0.109	0.101	419.6	445.6	466.8	500.6	516.9	544.6
50	0.193	0.245	0.174	0.094	0.112	0.094	433.8	464.6	491.1	516.3	547.6	577.2

Mean Message Transfer							Failure Rate					
σ/s 1	955	1128	1357	1544	1792	2141	0.000	0.000	0.000	0.000	0.000	0.000
2	1018	1313	1486	1788	1992	2218	0.000	0.000	0.000	0.000	0.000	0.000
3	1283	1561	1709	1817	2148	2469	0.000	0.000	0.000	0.000	0.000	0.000
4	1512	1716	1815	1993	2294	2614	0.000	0.000	0.000	0.000	0.000	0.000
5	1674	1857	2031	2399	2416	2570	0.004	0.000	0.000	0.000	0.000	0.000
6	1955	1971	2104	2299	2472	2660	0.004	0.000	0.000	0.000	0.000	0.000
7	1916	2182	2217	2529	2551	2886	0.000	0.000	0.000	0.000	0.000	0.000
8	2140	2214	2346	2650	2743	3157	0.004	0.000	0.000	0.000	0.000	0.000
9	2118	2322	2480	2742	3014	3074	0.000	0.000	0.000	0.000	0.000	0.000
10	2301	2425	2671	2984	2863	3184	0.000	0.000	0.000	0.000	0.000	0.000
15	2725	3182	3201	3512	3592	3969	0.008	0.000	0.000	0.000	0.000	0.000
20	3066	3336	3682	3723	4253	4570	0.008	0.004	0.004	0.000	0.000	0.000
25	3388	3952	4382	4401	4539	4866	0.004	0.000	0.000	0.000	0.000	0.000
30	4045	4341	4538	4934	4990	5366	0.004	0.000	0.000	0.000	0.000	0.004
35	4469	4756	4904	5109	5561	5837	0.016	0.004	0.000	0.000	0.000	0.008
40	4947	5083	5308	5723	5721	6356	0.000	0.004	0.012	0.004	0.004	0.008
45	5191	5512	5845	6234	6477	6919	0.004	0.008	0.008	0.000	0.012	0.012
50	5283	5680	6048	6389	6808	7238	0.004	0.004	0.012	0.020	0.032	0.024

As already mentioned above, the performance of the other collective decision-making strategies exhibits trade-offs at different parameter settings. The Pareto frontiers of the trade-offs between mean consensus time and mean absolute error in each strategy are shown in solid lines in Fig. 3.

DMVD's performance is shown using ∗ markers in Fig. 3, and its Pareto frontier is shown in magenta. It outperforms the Individual Exploration baseline, as it reaches the same error of 0. DVMD can reach a mean consensus time of 518 s and a failure rate of 0.1. The Pareto frontier of DMVD has a low gradient, and hence, it can obtain very low error values, but when the decision-making process needs to be faster, the error rapidly increases.

DC's performance is shown in ○ markers in Fig. 3, and its Pareto frontier is shown in red. DC is able to come to a consensus much faster and at much lower failure rate than Individual Exploration and DMVD. We observe that DC is able to produce a mean error of 0.256 at a mean consensus time of 145 s and a failure rate of 0, while DMVD requires a

Table 2 Mean absolute error, mean consensus time, mean message transfer and failure rate of DMVD at different σ and τ settings; vertical-σ mean lengths of exploration/dissemination states, horizontal-τ probability of mutating the chosen option
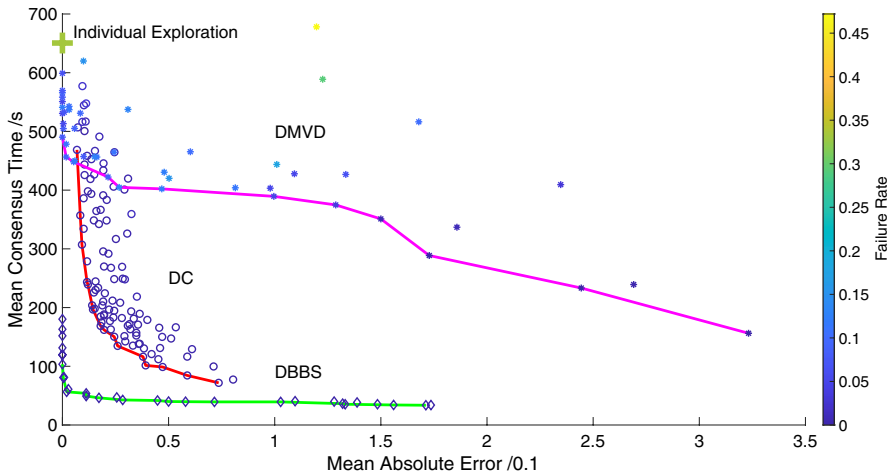
		Mean Error/0.1						Mean Consensus Time/s					
	τ	0.00	0.01	0.02	0.03	0.04	0.05	0.00	0.01	0.02	0.03	0.04	0.05
σ/s	1	3.232	2.691	2.347	1.679	1.226	1.197	156.2	239.3	409.4	516.3	588.8	678.1
	2	2.444	1.858	1.093	0.603	0.309	0.099	233.3	336.8	427.8	465.4	537.4	619.9
	3	1.728	0.996	0.479	0.243	0.084	0.031	288.7	389.4	430.5	464.5	531.0	542.6
	4	1.500	0.979	0.502	0.053	0.057	0.000	351.0	403.3	420.1	448.3	505.0	565.8
	5	1.288	0.468	0.215	0.018	0.032	0.000	374.8	402.2	422.3	456.4	537.1	541.0
	6	1.335	0.267	0.100	0.000	0.000	0.000	426.8	404.7	457.2	490.3	530.7	550.9
	7	0.814	0.162	0.059	0.004	0.004	0.000	404.0	456.4	450.0	504.8	513.3	569.6
	8	1.010	0.151	0.018	0.009	0.000	0.000	443.8	456.8	478.3	532.7	558.3	599.0
		Mean Message Transfer						Failure Rate					
σ/s	1	1298	2011	3510	4520	5290	6151	0.000	0.004	0.032	0.104	0.292	0.472
	2	1169	1763	2463	2865	3420	4147	0.000	0.012	0.052	0.104	0.132	0.152
	3	1051	1644	2075	2435	2844	3134	0.000	0.048	0.124	0.112	0.100	0.108
	4	998	1454	1771	2190	2526	3051	0.008	0.052	0.156	0.092	0.092	0.096
	5	963	1417	1669	1979	2487	2723	0.040	0.120	0.088	0.092	0.116	0.120
	6	960	1468	1790	2059	2414	2680	0.068	0.132	0.080	0.080	0.052	0.052
	7	1046	1574	1733	2175	2261	2735	0.116	0.112	0.120	0.080	0.056	0.076
	8	973	1526	1808	2186	2478	2824	0.180	0.128	0.112	0.072	0.080	0.068

Table 3 Mean absolute error, mean consensus time, mean message transfer and failure rate of DBBS at different λ and μ settings; vertical-λ decay rate of past neighbors' beliefs, horizontal-μ decay rate of current neighbors' beliefs during communication

		Mean Error/0.1						Mean Consensus Time/s					
	μ	0.0	0.2	0.4	0.6	0.8	1.0	0.0	0.2	0.4	0.6	0.8	1.0
λ	0.5	0.000	0.000	0.008	0.172	0.580	1.320	180.3	119.8	81.1	46.2	39.7	36.3
	0.6	0.000	0.000	0.020	0.284	1.028	1.484	163.0	103.5	56.5	42.7	39.3	35.4
	0.7	0.000	0.004	0.112	0.500	1.280	1.560	151.9	80.9	49.3	40.0	39.9	34.1
	0.8	0.000	0.028	0.256	0.716	1.388	1.712	131.4	60.9	46.9	39.4	38.4	33.6
	0.9	0.000	0.112	0.448	1.096	1.332	1.736	119.2	53.9	41.7	40.6	35.3	33.9
		Mean Message Transfer						Failure Rate					
λ	0.5	1687	1126	769	447	386	355	0.000	0.000	0.000	0.000	0.000	0.000
	0.6	1525	977	542	414	383	347	0.000	0.000	0.000	0.000	0.000	0.000
	0.7	1425	768	474	389	388	335	0.000	0.000	0.000	0.000	0.000	0.000
	0.8	1236	584	452	383	374	330	0.000	0.000	0.000	0.000	0.000	0.000
	0.9	1121	518	405	395	346	333	0.000	0.000	0.000	0.000	0.000	0.000

consensus time of 415 s and a failure rate of 0.084 to produce a comparable error of 0.249. However, DC is not able to produce a mean error as low as DMVD. In our experiments, the lowest mean error achieved is 0.0569 at a mean consensus time of 525 s, which is higher than those produced by DMVD at lower consensus time. However, the failure rate is smaller than those produced by the two previous algorithms at 0.016. Tuning its parameters to extend the mean consensus time has an increasingly diminishing impact on reducing the error for DC, as shown by the increasingly vertical slope of the DC's Pareto frontier on the top-left side.

DBBS's performance is shown in ◇ markers in Fig. 3, and its Pareto frontier is shown in green. In contrast to Individual Exploration and DMVD, DBBS is able to reach an error of zero at a much smaller mean consensus time of 102 s and without any failures. This is well beyond the Pareto frontiers of both DMVD and DC. On the other hand, the reduction of consensus time to lower than 60 s causes a dramatic rise of error from close to 0 to

Fig. 3 Plot of mean consensus time against mean absolute error for all considered algorithms at all considered parameter settings in random environment; +—Individual Exploration, ∘—DC, *—DMVD, ⋄—DBBS, color coding of markers shows the failure rate; Pareto frontiers: red—DC, magenta—DMVD, green—DBBS
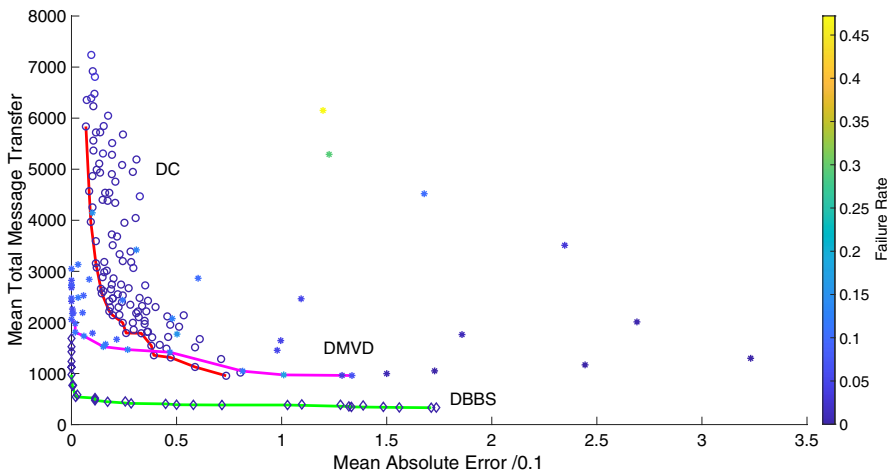
Fig. 4 Plot of mean total message transfer against mean absolute error for all considered algorithms at all considered parameter settings in random environment; ∘—DC, *—DMVD, ⋄—DBBS, color coding of markers shows the failure rate; Pareto frontiers: red—DC, magenta—DMVD, green—DBBS (Color figure online)

1.736. As shown in Table 3, these data points correspond to high values of the parameter μ. This is because at high settings of μ, the neighbors' beliefs are weighted heavily when computing an agent's own belief, and thus, strong positive feedbacks can occur; hence, a few agents can lead the swarm into making a premature consensus.

For all the considered collective decision-making algorithms, there are trade-offs between accuracy and consensus speed in their decision-making process. Regarding the

three benchmark algorithms, both DMVD and DC clearly outperform the Individual Exploration baseline, while the two collective strategies exhibit different characteristics in the accuracy vs speed trade-off. DMVD can reach higher accuracy, but is inelastic in terms of decision speed. DC, on the other hand, can form a consensus quickly, but is inelastic in terms of accuracy. In contrast, DBBS is able to outperform the state-of-the-art decision-making algorithms in all three metrics.

In Fig. 4, we plot the mean total number of message transfers against mean absolute error to investigate the performances of considered algorithms from the point of view of the communication overhead. Among the three algorithms, DMVD has the lowest average rate of message transfer at 4.96/s. This causes DMVD to surpass DC with most configurations in terms of communication required. DBBS has an average rate of message transfer at 9.53/s. This is close to the rate of message transfer intended in the experimental setup of one message per agent per 2 s, which produces an ideal rate of 10/s with 20 agents. The actual rate is likely to be lower because agents sometimes have no peers in their communication range. DC has the highest rate of message transfer at 12.69/s. This is because in our experiments DC, similar to DMVD, does not restrict the neighborhood size. Hence, agents can receive all messages sent from neighbors within its communication radius during a fixed time period. But in DBBS, agents are restricted to receiving only one message per control loop.

4.2 Concentrated environmental feature distribution

It has been observed in related works (Bartashevich and Mostaghim 2019; Shan and Mostaghim 2020) that higher concentration of black and white tiles can have a negative influence on the accuracy and speed of decision-making in the collective perception scenario. In other words, collective perception in an arena with random distribution of tiles is easier than in arenas where tiles of one color are all clustered together. In the discrete collective estimation scenario considered here, concentration of environmental features can cause the agents to have different fill ratio estimates from each other and, therefore, increase the importance of consensus forming in the decision-making process. We test the considered algorithms on situations with increasing concentration of black tiles. We randomly generate a series of environments as shown in Fig. 5 all with fill ratio of 0.45. We place a number of blocks of black tiles with a random width in the arena and add or take away individual tiles to reach the required fill ratio. The block widths are normally distributed with a mean set to reflect the desired level of concentration and standard deviation of 1. As before, 50 experimental runs are conducted for all environmental and design configurations.

We pick the settings shown in Table 4, which produce moderate results on the Pareto frontiers and test the performances of considered algorithms in a series of environments with progressively more concentrated distribution of black tiles. The changes in mean absolute error, mean consensus time and failure rate are plotted in Fig. 6. In this plot, a mean block width of 0 means that the tiles are distributed randomly.

In this figure, Individual Exploration's performances are shown in black. It can be observed that the mean error remains 0, when mean block width increases from 0 to 12, while the failure rate increases rapidly to close to 1. DMVD and DC both experience increase in mean error, mean consensus time and failure rate as the mean block width increases as shown in magenta and red colors, respectively. The increase in error
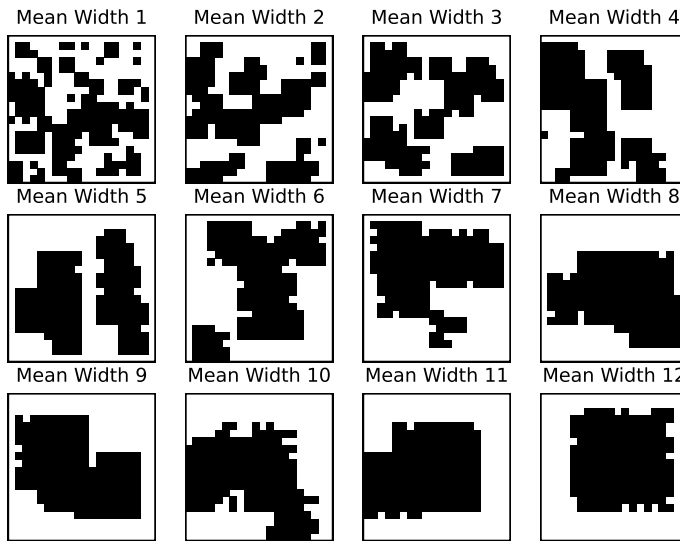
Fig. 5 Example of various arenas with different mean widths of blocks. Each square shows a randomly generated arena to be used in an experimental run. Black and white indicate the tiles of the respective colors. As the mean widths of blocks increase, the black tiles become more concentrated (Color figure online)

is of comparable degree. DC has a more significant increase in consensus time from 251 s to 399 s, and DMVD has a more significant increase in failure rate from 0.115 to 0.185. DBBS also experience a rise in error and a small rise in consensus time, but the failure rate remains zero even at the highest mean block width setting, as shown in green. This shows that DBBS's robustness remains even in scenarios with high concentration of environmental features.

To obtain a full picture of the performances of considered algorithms in more difficult environments, the performances of considered algorithms across all considered parameter settings in arenas with fill ratio of 0.45 mean block width of 12 are shown in Fig. 7. Individual Exploration has a failure rate close to 1 and therefore is not plotted. The Pareto frontiers obtained in the previous section in random environments are also shown in dotted lines to provide comparison.

It can be observed in the magenta line in Fig. 7 that there is a significant rise on the left-hand side of the Pareto frontier compared to in random environments. Thus, a lot more consensus time needs to be sacrificed to reach a low error. An error of 0 can only be reached by extending the consensus time to 809 s. There is also a large increase in failure rate, which reaches above 0.5 for many settings. This is because DMVD has a poor ability to ensure the formation of a consensus, compared to DC and DBBS. This shortcoming is more prevalent in an environment with high concentration of features, as the robots can hold conflicting opinions skewed to either extremes. Without enough pressure toward consensus, the probability of the robots to reach a consensus in the required time is low.

There is an increase in mean error and mean consensus time across all parameter settings for both DC and DBBS, compared to in the previous subsection, as shown in the red and green lines, respectively, in Fig. 7. DC has an even increase in both mean error and mean consensus time across all parameter settings. The failure rate also increases significantly when mean consensus time is high. On the top left end of the Pareto front, the

Table 4 Parameter settings of considered algorithms in Fig. 6

|		
DC	$\sigma = 8s$	$\tau = 0.05$
DMVD	$\sigma = 4s$	$\tau = 0.02$
DBBS	$\lambda = 0.7$	$\mu = 0.4$
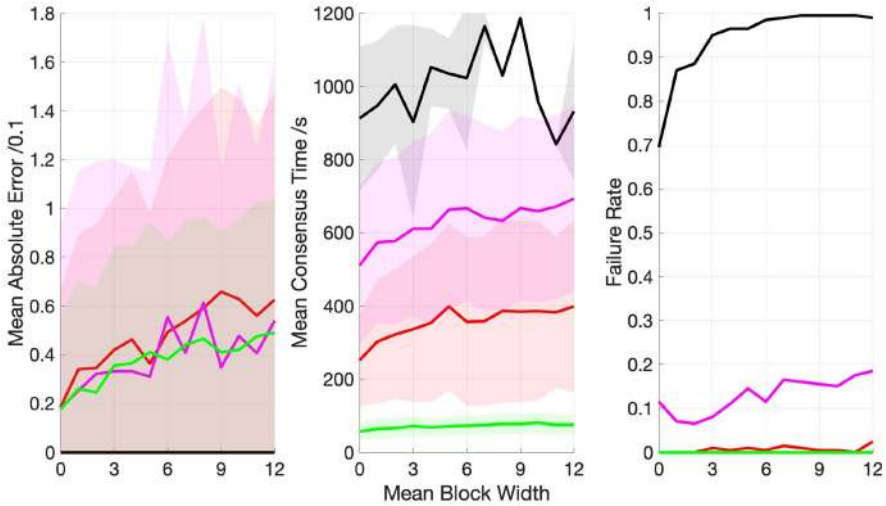
Fig. 6 Progression of performances of considered algorithms as the level of concentration of features increases; black—Individual Exploration, red—DC, magenta—DMVD, green—DBBS; shaded areas represent standard deviations (Color figure online)

minimum mean error DC can achieve increases from 0.069 to 0.4, while the mean consensus time increases from 469 s to 540 s, and the failure rate increases from 0.08 to 0.1.

DBBS has a high increase in error from 1.74 to 3.64 when mean consensus time is at its lowest, which is the range with high μ settings, but there is not a significant increase in consensus time in this section, only from 33.9 s to 37.5 s. This is because high level of positive feedback can guarantee that the robots would come to a consensus very quickly. However, this comes at the cost that robots could only take very few samples. In an environment with high concentration of features, these samples are likely to be skewed to one color and cause the final consensus to be inaccurate. At low μ settings on the top left end of the Pareto front, there is a mild increase in error and consensus time across most parameter settings. But zero error can still be achieved by extending the consensus time, which needs to be increased from 104 s to 357 s, while the failure rate can still be kept at 0.

4.3 Analysis of DBBS under sparse communications

In order to gauge the performances of our proposed DBBS algorithm under sparse communications, we vary the communication range of the robots and plot its effects on the performances of DBBS in environments with random feature distribution in Fig. 8. Besides the default communication range of 0.5 m, we also test 0.1 m and 0.3 m. All three cases result in no failure in decision-making, and therefore, the failure rate is not included in this figure.
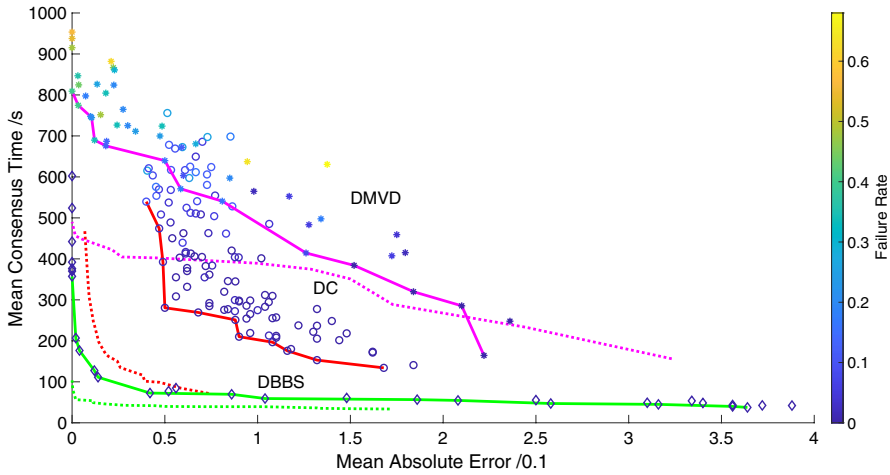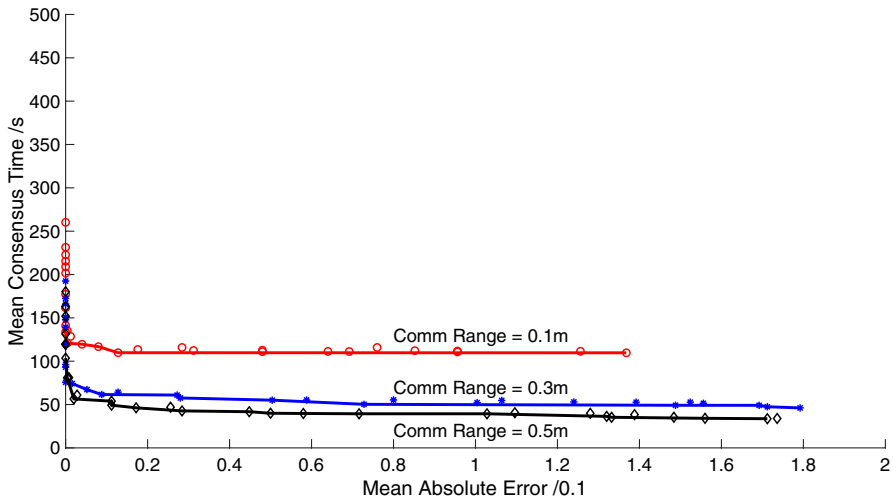
Fig. 7 Plot of mean consensus time against mean absolute error for all considered algorithms at all considered parameter settings in environment with fill ratio = 0.45, block mean width = 12; ∘—DC, ∗—DMVD, ◇—DBBS, color coding of markers shows the failure rate; Pareto frontiers: red—DC, magenta—DMVD, green—DBBS, dotted—Pareto frontiers in random environments (Color figure online)

When the communication range reduces from 0.5 m to 0.3 m, there is a mild increase in the consensus time of DBBS. When the communication range is reduced further to 0.1 m, the consensus time significantly increases to almost twice that of a communication range of 0.3 m. On the other hand, the general shapes of the Pareto frontiers between consensus time and error are largely the same. The algorithm is also able to reach an error of 0 for all communication ranges without much sacrifice of consensus time. This demonstrates that DBBS is resilient to the effects of sparse communications. A reduced communication range translates largely to longer consensus time, while the accuracy and reliability both remain high.

As a benchmark, the performances of DC at a reduced communication range are shown in Fig. 9. A shorter communication range of 0.1 reduces the number of neighbors an agent is able to communicate within the dissemination periods. Thus, it makes the decision process slower, but also improves the accuracy of the results. However, the increase in the consensus time is more significant than in DBBS, and DC is still outperformed in this situation.

The performances of the same communication range configurations in environments with concentrated feature distribution are shown in Fig. 10. It can be observed that the change of performance as communication range decreases is similar to the results in random environments. Due to the higher difficulty of decision-making in concentrated environments, the algorithm experiences some failures in decision-making when the communication range is at 0.3 m and 0.1 m. However, the failure rate is very low and not exceeding 0.08. Also, the failures arise in mostly outlying parameter configurations that are not on the Pareto frontier. For a communication range of 0.1, an error of 0 can still be achieved at a consensus time of 253 s, which is well beyond the Pareto frontiers of other considered decision-making strategies.
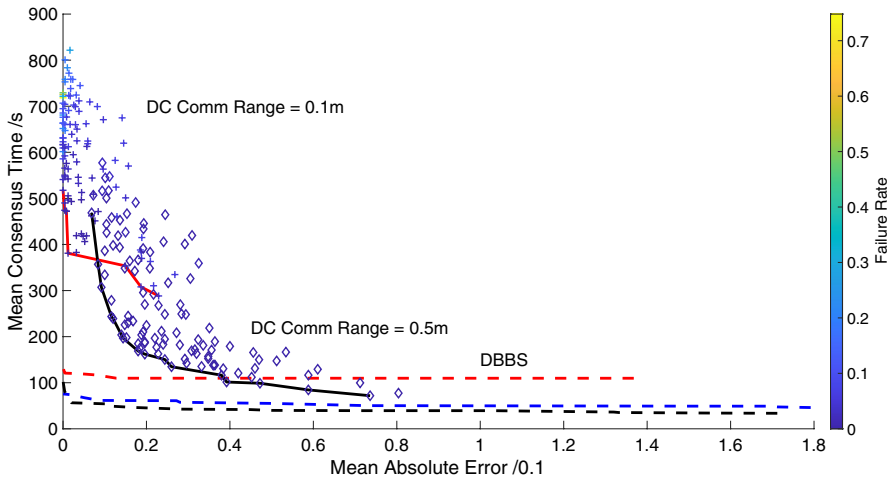
Fig. 8 Plot of mean consensus time against mean absolute error for DBBS with different communication ranges in random environment; red∘-0.1*m*, blue∗-0.3*m*, black⋄-0.5*m* (Color figure online)

5 Discussion

As demonstrated above, collective decision-making strategies all exhibit a trade-off between speed and accuracy in their parameter settings in operation. This is in line with the related literature (Brambilla 2015; Ebert et al. 2020; Shan and Mostaghim 2020). When applying collective decision-making strategies to scenarios with more than two options, the designers also need to contend with maintaining a diversity of opinions in the decision-making process for the agents to consider all of the available options without premature elimination, as well as coming to a consensus by the end of the run. This creates another trade-off on the design level between reliability and accuracy. This is demonstrated by the performances of DC, which is reliable but not accurate, and those of DMVD and Individual Exploration, which is accurate but unreliable.

It has been observed by Talamali et al. (2019) that, compared to stochastic switching methods, DC appears to be less accurate but faster, due to the higher amount of information transfer but also the propagation of errors across the swarm. We have observed a similar performance difference between DC and DMVD. From a bi-objective perspective, the Pareto frontier for DC tends to have an overall high gradient in a consensus time vs error plot, causing the error to be inelastic and hard to be reduced by slowing down the consensus speed. Since in our implementation of DC, quality measurements are not adopted by other robots, a similar error propagation does not exist. Instead, DC is still able to enforce a consensus via an elitist selection mechanism. For example, a single robot that obtains a high option quality can make every robot with which it is in contact, to change their decision. Thus, such a robot can possess a disproportionate amount of influence and can cause a premature convergence to a wrong option. In contrast, DMVD's Pareto frontier has an overall low gradient in a consensus vs error plot, causing the consensus time to be inelastic and not easily reduced by sacrificing accuracy.

DBBS is able to achieve better performances in terms of accuracy, speed and reliability. Its better performance comes at a cost of higher communication complexity. The

Fig. 9 Plot of mean consensus time against mean absolute error for DC with different communication ranges in random environment; red+—0.1m, black◊—0.5m, color coding of markers shows the failure rate. The Pareto frontiers produced by DBBS are shown in dashed lines (Color figure online)
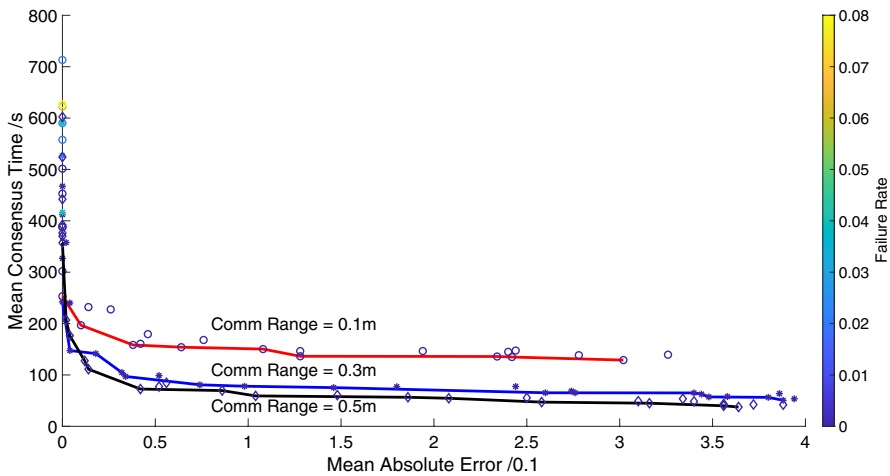
Fig. 10 Plot of mean consensus time against mean absolute error for DBBS with different communication ranges in environment with fill ratio = 0.45 block mean width = 12; red◦-0.1m, blue∗-0.3m, black◊-0.5m, color coding of markers shows the failure rate (Color figure online)

communication bandwidths required for DMVD and DC are both independent of the number of options. DMVD only needs the robots to transmit the chosen option, while DC needs the robots to transmit the chosen option and its corresponding quality. In contrast, DBBS requires the robots to transmit all estimated qualities. This causes the communication bandwidths required to scale with the number of potential options. In the scenario considered in this paper, the ratio of communication bandwidths required for DMVD, DC and DBBS is

roughly 1:2:10. Therefore, to implement DBBS on hardware, robots need higher communication bandwidths. On the other hand, DBBS has no need for robot identification, while in practice both DC and DMVD require the agents to broadcast a randomly generated 16-bit identifier to differentiate each other in a neighborhood (Valentini et al. 2016b).

DBBS's performance is also sensitive to its parameter settings. Particularly, a high μ setting can cause strong positive feedback and produce a very high error. This effect is further magnified when the algorithm is applied to an environment with a concentrated feature distribution. A potential solution is to implement a similar identifier as DC and DMVD and use the memory of past received messages to enable the agents to avoid logging messages from the same sender in a short period of time.

When comparing the performances of considered algorithms with their communication bandwidths required in mind, we can conclude that generally as the communication bandwidths increase between agents, there is a larger pressure on all agents to converge to a single consensus. This effect increases from Individual Exploration to DMVD and then to DC. This increased pressure for consensus has a positive influence on consensus speed and reliability, but can have a negative influence on the accuracy.

This shows the shortcoming of existing collective decision-making algorithms based on finite state machines, in which agents hold only one option and try to converge to the optimal option through dissemination. Our proposed algorithm DBBS avoids this shortcoming by having agents communicate with each other using the belief of the options. Thus, the agents can hold multiple likely options simultaneously during dissemination. This feature of DBBS ensures a high diversity of opinions among agents and also reliable convergence to an optimal option.

When applied to more difficult environments with more concentrated features, DC and DMVD exhibit a significant increase in error, consensus time and failure rate, while the performances of DBBS worsen mildly, with no increase in failure rate for most parameter settings. The increase in error can also be mitigated by parameter selection that slows the decision-making process. DBBS is also proven to be able to withstand the effects of sparse communications with only limited sacrifice of the consensus time.

When comparing our proposed DBBS algorithm with other related algorithms in collective decision-making, our approach shares much motivation with decision-making strategies based on opinion pooling (Lee et al. 2018a, b; Crosscombe et al. 2019). These strategies also attempt to perform multi-option collective decision-making by directly combining estimations of option qualities. Our approach differs firstly in that we design our algorithm with a physics-based environment in mind, such that we abandon the bidirectional communication required for opinion pooling and utilize a broadcast-based communication model. We also take advantage of the characteristics of the discrete collective estimation scenario and use a Bayesian statistics-based method to compute option qualities. Therefore, our algorithm has less communication and computational complexity than decision-making strategies based on opinion pooling.

The utilization of Bayesian statistics in our proposed DBBS algorithm is similar to the work by Ebert et al. (2020) and our previous work (Shan and Mostaghim 2020). Ebert et al. used a similar Bayesian approach in computing the option qualities. However, their algorithm is designed for a binary decision-making scenario. It is also intended to run on Kilobots, thus implying that the robots only engage in binary communications to share the current observations.

6 Conclusion

In this paper, we have extended collective decision-making to a novel discrete collective estimation scenario with more than two options. We have applied both direct comparison and direct modulation of voter-based decisions to the scenario using a Bayesian hypothesis testing method to compute option qualities. We compare their performances with those of an Individual Exploration baseline algorithm. We have also proposed a novel algorithm called distributed Bayesian belief sharing, to perform decision-making in the discrete collective estimation scenario.

We have tested the considered algorithms using different parameter settings in both environments with random distribution of features and environments with progressively more concentrated features. In the former environmental setting, we have concluded that both DC and DMVD outperform the Individual Exploration baseline. DC can also come to a consensus much faster than DMVD and Individual Exploration, while sacrificing some accuracy. Our proposed algorithm DBBS is able to reach a much better accuracy and speed performance but also maintain a failure rate of 0 in our experiments. As the features in the environment become more concentrated, Individual Exploration breaks down completely and is unable to come to a consensus in the required time limit. Both DC and DMVD exhibit a significant increase in error, consensus time and failure rate. DBBS has a milder increase in error and consensus time and is able to maintain a failure rate of zero for all parameter settings. We then tested DBBS under the same environments, but with lower communication ranges than default. We have observed that DBBS is able to operate well under sparse communications with some sacrifice to decision speed.

Among the existing collective decision-making strategies based on finite state machines, increasing the communication bandwidth between agents makes the decision-making process faster and more robust, but decreases the accuracy. Our proposed algorithm DBBS is able to overcome this trade-off and can achieve a better performance on all three metrics.

In future works, we will extend the belief sharing approach to best-of-n scenarios where the qualities of options need to be computed one at the time, such as site selection or collective search. In this paper, we took advantage of the unique characteristics of the discrete collective estimation problem and compute the option qualities of all available options on all individual robots. When it comes to scenarios where a single robot can only compute the option quality of a single option at a time, the belief exchange mechanism needs to be modified for the belief sharing approach to work. We also plan to investigate how to mathematically model the decision-making process of DBBS, so that its decision-making mechanisms can be better understood and applied to new scenarios.

References

Alanyali, M., Venkatesh, S., Savas, O., & Aeron, S. (2004). Distributed Bayesian hypothesis testing in sensor networks. In *Proceedings of the 2004 American Control Conference*, vol 6, pp. 5369–5374 vol. 6, https://doi.org/10.23919/ACC.2004.1384706

Bartashevich, P., & Mostaghim, S. (2019). Benchmarking collective perception: New task difficulty metrics for collective decision-making. In: Moura Oliveira P., Novais P., Reis L.P. (eds) Progress in Artificial Intelligence. EPIA 2019. Lecture Notes in Computer Science, Springer, Cham, pp 699–711, https://doi.org/10.1007/978-3-030-30241-2_58

Brambilla, D. (2015). Environment classification: an empirical study of the response of a robot swarm to three different decision-making rules. Master's thesis, Polytechnic University of Milan, Piazza Leonardo da Vinci, 32, 20133 Milano MI, Italy.

Camazine, S., Franks, N. R., Sneyd, J., Bonabeau, E., Deneubourg, J. L., & Theraulaz, G. (2001). *Self-Organization in Biological Systems*. USA: Princeton University Press.

Crosscombe, M., Lawry, J., & Bartashevich, P. (2019). Evidence propagation and consensus formation in noisy environments. In: Ben Amor N., Quost B., Theobald M. (eds) Scalable Uncertainty Management. SUM 2019. Lecture Notes in Computer Science, Springer, Cham, vol 11940, pp 310–323, https://doi.org/10.1007/978-3-030-35514-2_23

Ebert, J. T., Gauci, M., Mallmann-Trenn, F., & Nagpal, R. (2020). Bayes bots: Collective Bayesian decision-making in decentralized robot swarms. (pp. 7186–7192). https://doi.org/10.1109/ICRA40945.2020.9196584

Garnier, S., Combe, M., Jost, C., & Theraulaz, G. (2013). Do ants need to estimate the geometrical properties of trail bifurcations to find an efficient route? a swarm robotics test bed. *PLOS Computational Biology, 9*(3), 1–12. https://doi.org/10.1371/journal.pcbi.1002903.

Hoballah, I., & Varshney, P. (1989). Distributed Bayesian signal detection. *IEEE Transactions on Information Theory, 35*(5), 995–1000. https://doi.org/10.1109/18.42208

Lee, C., Lawry, J., & Winfield, A. (2018a). Combining opinion pooling and evidential updating for multi-agent consensus. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence, AAAI Press, IJCAI'18, p 347–353, https://doi.org/10.5555/3304415.3304465

Lee, C., Lawry, J., & Winfield, A. (2018b). Negative updating combined with opinion pooling in the best-of-n problem in swarm robotics. In: Dorigo, M., Birattari, M., Blum, C., Christensen, A.L., Reina, A., & Trianni, V. (eds) Swarm Intelligence. ANTS 2018. Lecture Notes in Computer Science, Springer, Cham, vol 11172, pp 97–108, https://doi.org/10.1007/978-3-030-00533-7_8

Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz., A., Magnenat, S., Zufferey, J.C., Floreano, D., & Martinoli, A. (2009). The e-puck, a robot designed for education in engineering. Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions 1(1):59–65, http://infoscience.epfl.ch/record/135236

Olfati-Saber, R., Franco, E., Frazzoli, E., & Shamma, J. S. (2006). *Belief Consensus and Distributed Hypothesis Testing in Sensor Networks* (pp. 169–182). Berlin Heidelberg, Berlin, Heidelberg: Springer. https://doi.org/10.1007/11533382_11.

Reina, A., Valentini, G., Fernández-Oto, C., Dorigo, M., & Trianni, V. (2015). A design pattern for decentralised decision making. *PLoS ONE, 10*(10), 1–18. https://doi.org/10.1371/journal.pone.0140950

Şahin, E. (2005). Swarm robotics: From sources of inspiration to domains of application. In: Şahin E, Spears WM (eds) Swarm Robotics. SR 2004. Lecture Notes in Computer Science, Springer Berlin Heidelberg, vol 3342, pp 10–20, https://doi.org/10.1007/978-3-540-30552-1_2

Scheidler, A., Brutschy, A., Ferrante, E., & Dorigo, M. (2016). The k-unanimity rule for self-organized decision-making in swarms of robots. *IEEE Transactions on Cybernetics, 46*(5), 1175–1188. https://doi.org/10.1109/TCYB.2015.2429118

Shan, Q. & Mostaghim, S. (2020). Collective decision making in swarm robotics with distributed Bayesian hypothesis testing. In Dorigo, M., Stützle, T., Blesa, M.J., Blum, C., Hamann, H., Heinrich, M.K., & Strobel, V. (eds) Swarm Intelligence. ANTS 2020. Lecture Notes in Computer Science, Springer, Cham, vol 12421, pp 55–67, https://doi.org/10.1007/978-3-030-60376-2_5.

Strobel, V., & Dorigo, M. (2018). Blockchain technology for robot swarms: A shared knowledge and reputation management system for collective estimation. In: Swarm Intelligence: 11th International Conference, ANTS 2018, Rome, Italy, October 29–31, 2018, Proceedings, Springer, vol 11172, p 425.

Strobel, V., Castelló Ferrer, E., & Dorigo, M. (2018). Managing byzantine robots via blockchain technology in a swarm robotics collective decision making scenario. In Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, AAMAS '18, p 541–549, https://doi.org/10.5555/3237383.3237464.

Talamali, M. S., Marshall, J. A. R., Bose, T., & Reina, A. (2019). Improving collective decision accuracy via time-varying cross-inhibition. In 2019 International Conference on Robotics and Automation (ICRA), pp 9652–9659, https://doi.org/10.1109/ICRA.2019.8794284.

Valentini, G., Hamann, H., & Dorigo, M. (2014). Self-organized collective decision making: The weighted voter model. In Proceedings of the 2014 International Conference on Autonomous Agents and Multi-Agent Systems, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, AAMAS '14, p 45–52, https://doi.org/10.5555/2615731.2615742.

Valentini, G., Hamann, H., & Dorigo, M. (2015). Efficient decision-making in a self-organizing robot swarm: On the speed versus accuracy trade-off. In Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, AAMAS '15, p 1305–1314, https://doi.org/10.5555/2772879.2773319.

Valentini, G., Brambilla, D., Hamann, H., & Dorigo, M. (2016a). Collective perception of environmental features in a robot swarm. In Dorigo, M., Birattari, M., Li, X., López-Ibáñez, M., Ohkura, K., Pinciroli, C., & Stützle, T. (eds) Swarm Intelligence. ANTS 2016. Lecture Notes in Computer Science, Springer, Cham, vol 9882, pp 65–76, https://doi.org/10.1007/978-3-319-44427-7_6.

Valentini, G., Ferrante, E., Hamann, H., & Dorigo, M. (2016b). Collective decision with 100 kilobots: Speed versus accuracy in binary discrimination problems. *Autonomous Agents and Multi-Agent Systems, 30*(3), 553–580. https://doi.org/10.1007/s10458-015-9323-3

Valentini, G., Ferrante, E., & Dorigo, M. (2017). The best-of-n problem in robot swarms: Formalization, state of the art, and novel perspectives. *Frontiers in Robotics and AI, 4*, 9. https://doi.org/10.3389/frobt.2017.00009

Varshney, P. K., & Al-Hakeem, S. (1991). *Algorithms for sensor fusion, decentralized Bayesian hypothesis testing with feedback* (Vol. 1). Kaman Sciences Corp Colorado Springs CO: Tech. rep.

Wang, L., & Xiao, F. (2010). Finite-time consensus problems for networks of dynamic agents. *IEEE Transactions on Automatic Control, 55*(4), 950–955. https://doi.org/10.1109/TAC.2010.2041610