

This is a repository copy of *A study of error diversity in robotic swarms*{*A study of error diversity in robotic swarms for task partitioning in foraging tasks.*

White Rose Research Online URL for this paper:

<https://eprints.whiterose.ac.uk/194309/>

Version: Accepted Version

Article:

Buchanan Berumen, Edgar orcid.org/0000-0001-6587-8808, Alden, Kieran James, Pomfret, Andy orcid.org/0000-0003-0325-1617 et al. (2 more authors) (Accepted: 2022) *A study of error diversity in robotic swarms*{*A study of error diversity in robotic swarms for task partitioning in foraging tasks.* *Frontiers in Robotics and AI.* ISSN 2296-9144 (In Press)

Reuse

This article is distributed under the terms of the Creative Commons Attribution (CC BY) licence. This licence allows you to distribute, remix, tweak, and build upon the work, even commercially, as long as you credit the authors for the original work. More information and the full terms of the licence here:

<https://creativecommons.org/licenses/>

Takedown

If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.

A study of error diversity in robotic swarms for task partitioning in foraging tasks

Edgar Buchanan^{1,*}, Kieran Alden¹, Andrew Pomfret¹, Jon Timmis², and Andy M. Tyrrell^{1*}

¹*School of Physics, Engineering and Technology, University of York, UK*

²*School of Computer Science, University of Sunderland, UK*

Correspondence*:

Corresponding Author

edgar.buchanan@york.ac.uk

2 ABSTRACT

3 Often in swarm robotics, an assumption is made that all robots in the swarm behave the same
4 and will have a similar (if not the same) error model. However, in reality this is not the case and
5 this lack of uniformity in the error model, and other operations, can lead to various emergent
6 behaviours. This paper considers the impact of the error model and compares robots in a swarm
7 that operate using the same error model (uniform error) against each robot in the swarm having a
8 different error model (thus introducing error diversity). Experiments are presented in the context
9 of a foraging task. Simulation and physical experimental results show the importance of the error
10 model and diversity in achieving expected swarm behaviour.

11 **Keywords:** swarm robotics, fault-tolerance, error diversity, task partitioning, foraging

1 INTRODUCTION

12 Robot swarms are capable of performing different tasks in an efficient and decentralized way, in part, due
13 to their high level of parallelization. In the past, swarm robotic systems were assumed to be inherently
14 robust to failures due to their high degree of robot redundancy. However, Winfield and Nembrini (2006)
15 demonstrated that this is not always the case, and that specific processes must be introduced to increase the
16 robustness of the swarm. Bjercknes and Winfield (2012) demonstrated that in various scenarios, specific
17 partial failures in robots can lead to task degradation or, in the worst case, the task not being achieved.

18 Fundamental mechanisms to achieve robust fault-tolerant swarm robotic systems, have been considered
19 by different authors including detecting faults (Tarapore et al., 2015; Christensen et al., 2009), diagnosing
20 the fault type (Keeffe et al., 2017; Li and Parker, 2007) or recovering from the fault (Humza et al., 2009;
21 Timmis et al., 2010), or a combination of all of these fault-tolerant methods (Parker, 1998; Parker and
22 Kannan, 2006). However, in all of these works, authors classify the robots according to their behaviour i.e.
23 a robot being either faulty or non-faulty, where faulty robots exhibit abnormal behaviour compared with
24 non-faulty robots. However, in reality, it is often hard to distinguish between a faulty and non-faulty robot
25 due to the diversity of errors across the swarm.

26 The importance of considering error diversity in swarm robotics can be summarized in three points. 1) It
27 is possible to reduce the reality gap between swarm behaviours in simulation and the swarm behaviours in
28 hardware. 2) The error non-diversity could lead to false positive results, as well as missing critical faults

29 that ultimately led to failures. 3) It is important for each robot to learn and adapt to its own inherent error in
30 this way, the robot swarm will exhibit better performance.

31 To the best of the authors' knowledge, there is no literature where diversity in terms of levels of error
32 is considered in fault tolerance experiments. Most related research examines diversity across the swarm
33 from an evolutionary perspective, where controllers can be evolved independently for each robot in the
34 swarm. For example, reinforcement learning has been used to train task-specialized robots where each
35 robot in the swarm learns to perform a specific task (Balch, 1998; Li et al., 2003; Balch, 2005) or tasks have
36 been previously allocated to robot members (Zhang et al., 2008). Robot controllers are generally evolved
37 with two methods: genetically homogeneous or genetically heterogeneous (Bongard, 2000; Mitchell A.
38 Potter, Lisa A. Meeden, 2001; Trianni and Nolfi, 2011; Tuci, 2014; Hart et al., 2018). In the homogeneous
39 method, the controller is expressed as a single genotype which is then cloned to each robot in the swarm.
40 With respect to the heterogeneous method, each robot has its own genotype and after evolution, each
41 robot has a specific role in the task. Despite this work on controller-heterogeneous swarm systems, there
42 is no research that considers the appropriate level of error for each robot during the simulation and its
43 impact on the performance of the swarm in simulation and in hardware. In this paper, we identify and
44 study the discrepancy of results when degrees of diversity of error are considered, referred in this paper as
45 *heterogeneous error* or if all the robots share the same degree of error they are referred to *homogeneous*
46 *error*.

47 The task considered in this paper is task partitioning in foraging. Task partitioning, first observed in
48 biological systems (Ratnieks and Anderson, 1999), has been used in the swarm robotics context to prevent
49 bottlenecks close to the home area where the items are deposited Goldberg and Matarie (2001); Brutschy
50 et al. (2014); Pini et al. (2011). Other approaches such as Pini et al. (2013, 2014); Buchanan et al. (2016)
51 have focused on fault-tolerance in order to increase performance when the dead-reckoning error is present
52 in a robot. Task partitioning has been also used to study the effect of task decomposition on emergent
53 swarm intelligence (Harwell et al., 2020).

54 Task partitioning is a technique that consists of dividing a single task into multiple smaller subtasks, with
55 the objective to reduce the amount of distance travelled by each robot and hence decreasing the error in
56 dead reckoning. A robot finds an item in the environment and transports the item towards the home area
57 for a short distance referred to as Partition Length (P). Then, the item is exchanged by either leaving the
58 item on the floor for a different robot to collect (indirect transfer) or waiting for the second robot to receive
59 the item directly from the first robot (direct transfer). Since the robots are travelling a shorter distance P
60 compared to the total distance between the home area and the items source, the dead-reckoning is smaller.
61 The amount of distance P depends on the approach taken.

62 The experimental framework in this paper follows a top-down approach from the macro to the micro
63 perspective, combined with three layers of abstraction: emulation, simulation and hardware.

64 In the first stage, emulation, an ensemble of different machine learning techniques is trained with a
65 dataset generated from latin-hypercube sampling in simulations. The main advantage of using emulation is
66 to save time and computation resources, compromising the resolution of the simulation. In other words, the
67 emulation is used to explore the experiments from a macro or global perspective which is only concerned
68 with the behaviour of the swarm as a whole. An emulation is generated for each strategy with each error
69 type.

70 The second and third stages, simulation and hardware, are studies from the micro or local perspective
71 which is concerned with the contribution of each robot to the task. This aids in obtaining a comprehensive
72 understanding of the behaviour of the swarm.

73 The contributions are summarized as follows:

- 74 • demonstration that the assumption made that all the robots in a simulated robotic swarm shared the
75 same error model can lead to unexpected swarm behaviours when testing the behaviour with physical
76 robots where each robot experiences a different error.
- 77 • difference in behaviours introduced by the error diversity can be mitigated by having each robot learn
78 the task according to the success of its performance, thus reducing errors for all robots.

79 The hypotheses explored in this paper are as follows:

- 80 1. the consideration of error diversity leads to different correlation values than non-diversity.
- 81 2. each robot adapts differently to its inherent error.

82 The rest of the paper is organized as follows. In section 2, a case study is presented and each task partitioning
83 strategy is described. The methodology followed in the experiments for this paper is described in section 3.
84 Experiments and comments on results are presented in section 4, and section 5 provides the summary and
85 conclusions.

2 CASE STUDY: FORAGING TASK PARTITIONING STRATEGIES

86 Foraging with dead-reckoning as navigation is used as a task to study the impact of the error diversity and
87 uniformity across the robots in the swarm to the behaviour and performance of the swarm.

88 Within a simple foraging task, a group of robots explore the environment searching for items to collect.
89 Then, after an item is found, the position is recorded by the robot and finally, the robot transports the item
90 towards the home area. For this paper single home area, however, this work could be used for multiple
91 places home (Lu et al., 2018).

92 Due to systematic and unsystematic errors in dead-reckoning navigation, the real-time position is often
93 inaccurate leading to an error in the positioning estimation. Error accumulates as a robot travels, which in
94 turn affects the estimated item position. As a consequence, when the robot attempts to go back to where it
95 found the last item, in order to collect further items, the robot reaches a different location. One simple way
96 to mitigate this error is to partition the distance travelled by each individual robot.

97 In this paper, two task partitioning foraging strategies are used to examine the effect of error diversity
98 and uniformity. The objective of this approach is to compare traditional foraging with a strategy where
99 robots learn to divide the task into multiple smaller tasks, depending on the success, or otherwise, of item
100 collection. Different emergent behaviours are expected to appear when error diversity and uniformity are
101 considered for each strategy. In the first part of this section the foraging task is described and the first
102 strategy, Non-partitioning Strategy (NPS) (Pini et al., 2011), is introduced. A second strategy is then
103 described Dynamic Partitioning Strategy (DPS) (Buchanan et al., 2016), where the number of partitions is
104 defined by a penalty and reward mechanism.

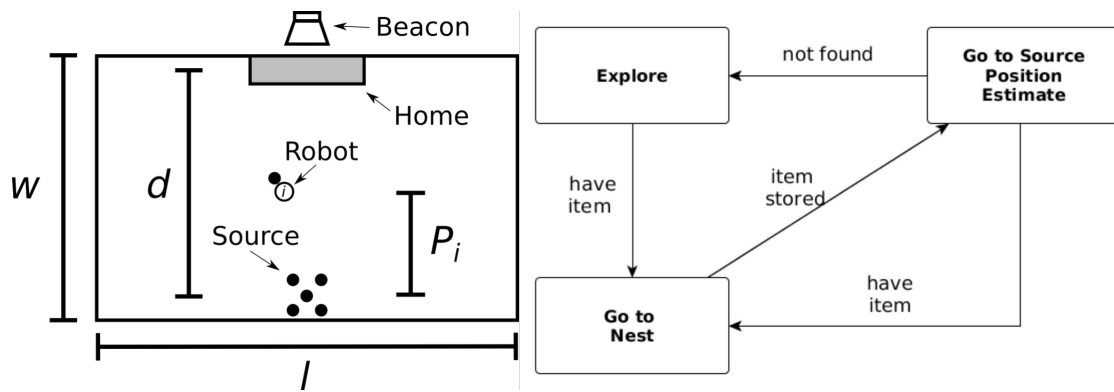


Figure 1. Foraging arena (left) and finite state machine used by the robots (right). Left: the arena has a rectangular shape defined by the width (w) and length (l). The distance between home area and items source is represented by d . P_i represents the individual partition length for robot i . Right: Non-partitioning strategy finite state machine composed of three states: *explore*, *go to nest* and *go to source*.

105 2.1 Foraging task and non-partitioning strategy

106 In both strategies, contained in the environment is a virtual beacon which guides the robots towards the
 107 home area (nest). The arena, shown in Figure 1, is rectangular and is defined by the width (w) and length
 108 (l) and the distance between the home area and the items to be collected (source (d)).

109 In the Non-partitioning Strategy (NPS) the robots take the items directly from the item source to the
 110 home area. The controller is a Finite State Machine (FSM) composed of three states as shown in Figure 1
 111 and each state is now described.

112 First, the robot starts in the *explore* state in which all the robots are searching for items in the arena. The
 113 exploration consists of a motor schema-based navigation (Arkin, 1987) with two behaviours: (i) stay on the
 114 path and (ii) avoid obstacles. Once an item is within the range of vision of the robot, the robot records the
 115 position, picks up the item and then enters the *go to nest* state.

116 In the *go to nest* state, the robot travels towards the home area by following the virtual beacon. The robot
 117 aligns itself towards the source of the light and moves straight to that location. The robot knows it has
 118 reached the home area when ground sensors on the robot detect a change in colour. The item is deposited,
 119 and the robot transitions to the *go to source* state.

120 The *go to source* state consists of the robot travelling back to where the estimated position of the items
 121 source is recorded. The robot uses dead-reckoning as navigation, to guide itself to reach the items source.
 122 If an item is found the robot picks up another item and switches to *go to nest* state, if the robot does not
 123 find an item then it changes to *explore* state.

124 In an ideal scenario, the robot would be continuously retrieving items from the source without transitioning
 125 to the *explore* state. However, in reality, robots are susceptible to systematic and non-systematic errors
 126 introduced by the dead-reckoning noise, which in turn, affects target position estimation. The error
 127 accumulates the longer the distance a robot travels, therefore, the probability of finding the items source
 128 decreases as d increases due to the drift from the actual items source and the estimated item's source
 129 position.

130 In order to decrease the error introduced by dead-reckoning noise, the task can be decomposed into
 131 smaller sub-tasks performed by each robot and this is described in the following section.

132 2.2 Dynamic partitioning strategy

133 In Dynamic Task Partitioning (DPS), each robot i changes its individual distance travelled (P_i) using a
 134 penalty and reward mechanism. Every time a robot finds an item in the *go to source* state P_i is increased. If
 135 the robot i does not find an item then P_i is decreased.

136 P_i is calculated with Equation 1 where k defines the amount of distance changed to P_i and α defines the
 137 ratio between reward and penalty. As α decreases, the robot is rewarded. $P_i(t)$ is the partition length after
 138 the application of the penalty and rewards, and $P_i(t - 1)$ is the value beforehand.

$$P_i(t) = \begin{cases} P_i(t - 1) + k(1 - \alpha) & \text{if item found} \\ P_i(t - 1) - k\alpha & \text{if item not found} \end{cases} \quad (1)$$

3 METHODOLOGY

139 This section presents the methodology followed for all experimental work in the paper. First, parameters
 140 and outputs are defined, then, the experimental framework followed and tools used are introduced and
 141 finally, the statistical techniques used to analyse the experiments results are described.

142 3.1 Terms, parameters and outputs

143 In order to understand the impact of having different error models for each robot in a swarm, compared
 144 to all robots having the same error model, it is important to perform a study from the macro (the swarm as
 145 a whole) and the micro (each robot by itself) perspectives.

146 From a macro perspective, the *total items collected* output provides a good performance metric for the
 147 swarm as a whole and this can be compared with different *swarm sizes* and distances between home arena
 148 and items source. The *social entropy* provides a metric for how homogeneous the swarm is, according to
 149 the different individual errors.

150 From a micro perspective, it is important the outputs reflect the performance of each robot by itself. The
 151 *collection ratio* output represents how successful a robot is at finding the item, providing an indication of
 152 how bad the error is for each robot. In a similar way, the *explore ratio* provides an indication of the amount
 153 of time the robot spends exploring.

154 3.1.1 Terms and parameters

155 To allow appropriate results to be collected and meaningful analysis to be undertaken, a number of
 156 environmental and system parameters must be defined for the experiments. The terms that are used for the
 157 experiments are *simulation length*, *swarm size* and the parameters are d , P and α .

158 *Terms:*

159 The *experiment length* term represents the duration of the experiment until it stops.

160 The *swarm size* term represents the total number of robots performing the task. The minimum *swarm size*
 161 is 2 because at least a pair of robots are required to have task partitioning. The maximum *swarm size* is 15
 162 as with greater values the robots spend more time avoiding each other than collecting items for the given
 163 environments.

164 *Parameters:*

165 The distance between the home area and the items source is shown as d .

166 *Swarm density* is an implicit parameter explored in this paper which is correlated to the swarm size and d .

167 The *partition length* (P) parameter represents the distance that a robot travels from where it found an
168 item for the first time, towards the home area, measured in meters. All robots start with the same P .

169 The α parameter regulates the amount of penalty and reward assigned to each robot in the swarm for
170 DPS.

171 3.1.2 Outputs

172 The outputs collected and subsequently used to compare and contrast the various methods from the
173 experiments are *final \tilde{P}* , *total items collected*, *explore ratio*, *collection ratio* and *social entropy*. These
174 outputs are described next.

175 The *total items collected* output represents the number of items collected at the end of the experiment.

176 P_i The *final \tilde{P}* represents the last median of \tilde{P} of all from all the robots when the experiment stops from a
177 uni-modal distribution.

178 The *explore ratio* represents the amount of time spent by the swarm in the *explore* state. The *explore ratio*
179 is measured as $\frac{T_E}{T_T}$ where T_E represents the sum of total time spent by all the robots undertaking iterations
180 in the *explore* state and T_T is the *simulation length* multiplied by the *swarm size*.

181 The *collection ratio* isolates the frequency a robot successfully collects an item allowing for the
182 identification of a correlation between the level of error and how often a robot retrieves an item. $\frac{I_F}{(I_F+I_L)}$.
183 I_F is a counter that records the number of times a robot transitions to the *go to nest* state from the
184 *neighbourhood exploration* state. I_L is a counter that records the number of times a robot enters the *explore*
185 state from the beginning of the simulation.

186 *Social entropy* is a metric that measures diversity in robot swarms and was initially introduced by Balch
187 (1998). This metric is used to measure robot homogeneity across the swarm according to the classification
188 of robots by their individual performance, where a robot can classify as faulty (high error) or non-faulty
189 (low error). Social entropy $H(R)$ is calculated with Equation 2 where M is the number of subsets (faulty
190 and non-faulty robots), p_i is the proportion of agents in each subset i and R represent the group of robots.
191 The lower value of $H(R)$ the more homogeneous the swarm is. This parameter output is explored in more
192 detail in Section 4.2.2.

$$H(R) = - \sum_{i=1}^M p_i \log_2(p_i) \quad (2)$$

193 3.2 Experimental framework and tools

194 The experimental framework is a top-down approach divided into three stages, as illustrated in Figure 2.
195 The first stage consists of studying the effect of implementing heterogeneous and homogeneous errors
196 from a macro perspective (considering the swarm as a whole) by performing a sensitivity analysis on
197 each strategy, via emulation. The second stage consists on studying the effects of heterogeneous and
198 homogeneous error from a local perspective (considering each robot as an individual) in simulation. Finally,
199 experiments with physical robots validate the results from emulation and simulation.

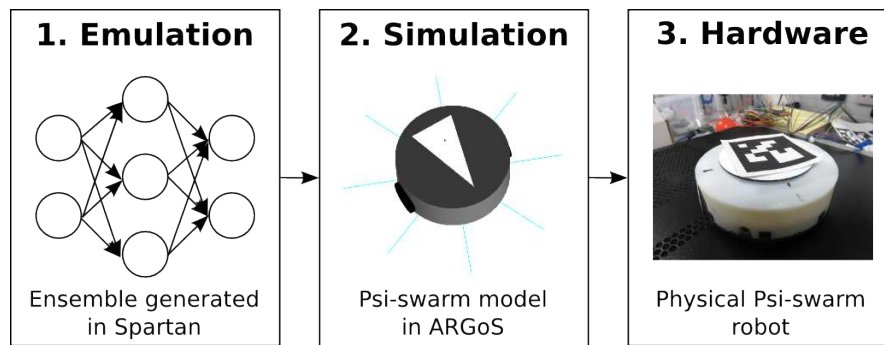


Figure 2. Framework for the experiments in this paper. 1) An emulation is generated from different machine learning techniques in order to perform an enriched model analysis from a macro perspective which could not be possible to do in simulations due to constraints in time, battery span and computational power. 2) Simulations are used to study the model from a micro perspective. 3) Experiments in hardware are used to validate results shown in emulation and simulation.

200 3.2.1 Simulation and hardware

201 The simulator used throughout the work described in this paper was ARGoS (Pinciroli et al., 2011),
 202 selected due to the support provided to run experiments for large numbers of robots. For the experiments in
 203 this paper, a simulated and real version of the psi-swarm robot platform (Hilder et al., 2014) (Figure 2).
 204 This robot has an infra-red and ground colour detector sensors. Since this robot does not have a camera,
 205 a virtual camera is used in hardware experiments instead to aid the robot to detect the items. The virtual
 206 camera consists of a tracking system that retrieves the position of an ArUco tag (Garrido-Jurado et al.,
 207 2014) attached on top of each robot. A blue-tooth signal is sent to the robot to let it know that it has found
 208 an item.

209 The source code for the psi-swarm controller used in the experiments detailed in this paper can be found
 210 online at https://github.com/edgarbuchanan/psiswarm_task_partitioning.

211 3.2.2 Training data

212 A total of 4 datasets are generated from the outputs of the latin-hypercube sampling (Mckay et al., 1998)
 213 in simulations for each strategy (NPS and DPS) and for each error type (heterogeneous and homogeneous).
 214 On average, 5 hours of simulated time represents roughly 1 minute in real-time. The number of replicates
 215 needed for the experiments shown in this paper is 180 (see Section 3.3 for more information). Therefore,
 216 for a set of experiments for a single strategy, it would take 3 hours in real-time. The amount of time required
 217 for experiments escalates if a population of samples and/or number of generations is required. Therefore,
 218 for parameter analysis that requires a large number of sample, the data set can take a long time to produce.

219 3.2.3 Emulator

220 Incorporating a combination of machine learning algorithms, an emulation is created that can be used as
 221 a surrogate for original simulations. This emulator is capable of making efficient predictions of simulation
 222 output for a given parameter set, reducing the time and resource requirements inherent in simulation due to
 223 the large number of replicates and size of the parameter space.

224 Parameters and outputs considered for the training of each emulator can be found in Tables 1.

225 The procedure to generate the emulator and use this to perform a predicted sensitivity analyses is as
 226 follows. For each parameter in each strategy, a value range is assigned and sampled using Latin-Hypercube

Table 1. Parameters and outputs used for the latin-hypercube sampling

Parameters		
Strategy	Name	Interval
NPS	<i>Swarm size</i>	[2-14]
	<i>d</i>	[0.5-2.0 m]
DPS	<i>Swarm size</i>	[2-14]
	<i>d</i>	[0.5-2.0 m]
	α	[0-1]
Outputs		
	Name	Interval
	<i>total items collected</i>	[0 max]
	<i>explore ratio</i>	[0 1]
	<i>collection ratio</i>	[0 1]

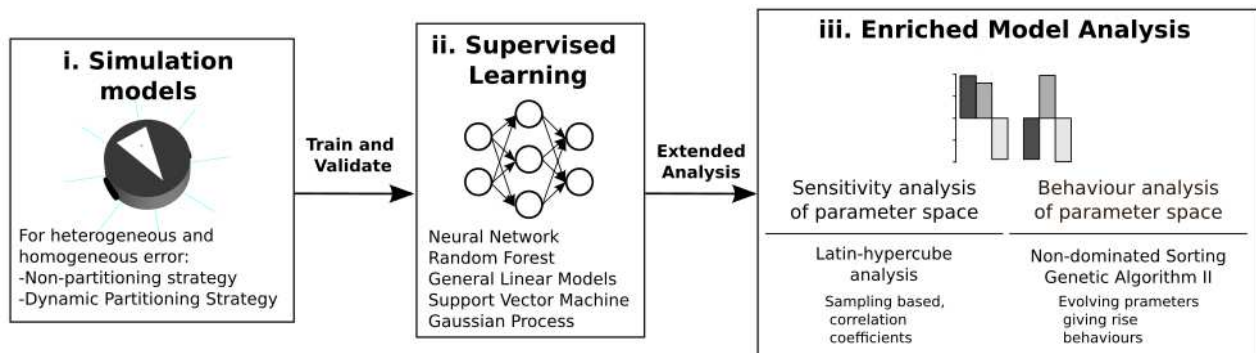


Figure 3. From simulation to emulation block diagram. A data set is generated from latin-hypercube analysis in simulations (i). This data set is used to train and validate the ensemble composed with different machine learning techniques (ii). Different statistical tools are used to analyse the ensemble model (iii).

227 Sampling, which ensures adequate coverage of the parameter space (Figure 3i). Then, each of the four
 228 datasets is used to train and validate the performance of five machine learning techniques (Neural Network,
 229 Random Forest, General Linear Models, Support Vector Machine and Gaussian Process).

230 These five individual emulators are combined to form one predictive emulation, or ensemble, where
 231 predictions are generated by weighting the performance of each algorithm on a test set (Figure 3ii).
 232 Combining the five algorithms has been shown to increase the accuracy of prediction over using each
 233 emulator in isolation (Alden et al., 2018).

234 Finally, the emulation is used to perform an enriched sensitivity analysis of the parameter space
 235 (Figure 3iii). Using sensitivity analyses, important parameters are revealed for each strategy, the
 236 understanding of the relationships between parameters and outputs with each is increased and regions for
 237 the parameters where maximum and minimum performance can be found.

238 The statistical tools used to provide a better understanding for each strategy are the following. To
 239 assess the degrees of dependency between parameters and outputs, Partial Rank Correlation Coefficients
 240 are calculated for each parameter-output response pair (Mckay et al., 1998). To determine whether the
 241 parameters can be optimised to produce the desired behaviour, the evolutionary algorithm Non-dominated
 242 Sorting Genetic Algorithm II (NSGA-II) (Deb et al., 2002) has been used. More information for each tool
 243 can be found in Section 3.3.

244 In this paper, we have included figures that show key results from these analyses, which are then
 245 discussed in more detail. However, for completeness, we include the results of the emulation training
 246 and test procedures and all statistical analyses in the supporting website [https://www.york.ac.uk/
 247 robot-lab/taskpartitioning/](https://www.york.ac.uk/robot-lab/taskpartitioning/).

248 3.3 Statistical tests and techniques

249 This section shows the statistical tests used to analyse the results from the experiments. The statistical
 250 analysis is performed using Spartan (Alden et al., 2013) for more information about each technique, please
 251 refer to (Alden et al., 2014) and (Alden et al., 2018).

252 *1) Consistency analysis:* The consistency analysis technique allows the identification of the number of
 253 executions that minimizes the effect of aleatory uncertainty caused by inherent stochastic variation within
 254 non-deterministic simulations. In this technique, 20 distributions are compared with the Vargha-Delaney
 255 test for a different number of runs. The Vargha-Delaney A test is a non-parametric effect magnitude test
 256 that can be used to indicate the difference between two distributions (Delaney and Vargha, 2000). The more
 257 different the distributions are, the closer the score is to 0 and 1. We believe that our data does not need
 258 to transformed as suggested in Neumann et al. (2015) because the amount of time that it takes for these
 259 variables to change is greater than the length of the tick used for simulations (0.1s). The value of 180 as
 260 sample size has an A-Test score below 0.56 which suggests that the aleatory uncertainty in the outputs has
 261 been mitigated and also avoids over-fitting the experiments with a larger sample size. In summary, this
 262 technique is used to minimize variation from non-determinism in the results to get the correct interpretation
 263 of the results.

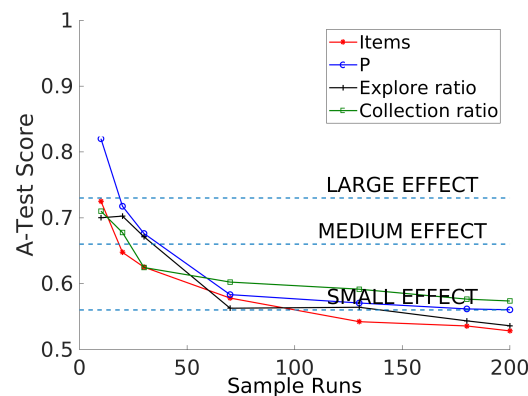


Figure 4. Consistency analysis that shows that the sample size value of 180 is large enough to avoid aleatory uncertainty effect (A-test lower than 0.56) in the outputs (item collected, P , explore ratio and collection ratio).

264 *2) Partial Rank Correlation Coefficients:* The Partial Rank Correlation Coefficients (PRCC) are used
 265 here in order to identify the degrees of dependency between each parameter. This is done by performing a
 266 Latin-hypercube sampling across parameters space and the number of samples is 1000. The main difference
 267 between random sampling and Latin-hypercube sampling is that with the latter, it is possible to increase
 268 reliability that the entire space is covered adequately. PRCC provides a measure of the influence of a single
 269 parameter with a single output. Strong correlations (close to 1 or -1) correspond to influential parameters
 270 over their respective output in spite of non-linearity introduced by other parameters. In summary, this

271 technique allows us to identify the key parameters for specific outputs and in this way, it is possible to
 272 identify if the error diversity or lack of it has any influence on these key parameters.

273 3) *Non-dominated Sorting Genetic Algorithm II*: Non-dominated Sorting Genetic Algorithm II (NSGA-II)
 274 is an evolutionary technique that explores the entire parameter space in order to maximize and/or minimize
 275 multiple outputs. Once, all the solutions population converge (or meet the specific convergence criteria)
 276 this set of solutions is referred to as Pareto front. In later sections of this paper, it will be revealed that the
 277 consideration of error diversity distorts this Pareto front and this is correlated to the number of high-error
 278 robots.

279 3.4 Error model

280 The error model consists of adding noise to each motor as shown in Equations 3 and 4. The simulated
 281 noise μ is generated from taking a sample from a Gaussian distribution each time tick with median $k[t]$
 282 and σ where σ changes for each robot. Curvature functions k for each robot can be found in the Appendix.
 283 Every time a robot changes the speed of its motors, timer t is set to 0. This noise model recreates the bias
 284 in the robot of moving towards a single direction. Examples of trajectories with physical robots are shown
 285 in Figure 5.

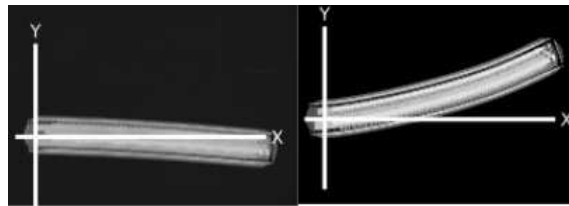


Figure 5. Trajectories of two different physical robots, A (left) and F (right), moving forwards for 30 seconds with a speed of 0.02 m/s.

286 Experiments recorded with physical robots and trajectories produced in the simulator for the first 6 robots
 287 are shown in Figure 6. The error model in simulation closely resembles the error model in hardware. A
 288 model of the psi-swarm used in the simulator in this work can be found in https://github.com/edgarbuchanan/psiswarm_model.
 289

$$\text{rightWheelSpeed} = \text{actuatedRightWheelSpeed} \pm \mu * \text{actuatedRightWheelSpeed} \quad (3)$$

$$\text{leftWheelSpeed} = \text{actuatedLeftWheelSpeed} \pm \mu * \text{actuatedLeftWheelSpeed} \quad (4)$$

290 It is important to mention that for the heterogeneous error, the error model for each robot is fixed and it
 291 does not change for the experiment shown in this paper.

4 EXPERIMENTS AND RESULTS IN A MULTI-SCALE MODEL APPROACH

292 This section presents experiments to demonstrate the performance difference between heterogeneous
 293 and homogeneous errors. First, by using emulation, a sensitivity analysis is performed for each strategy

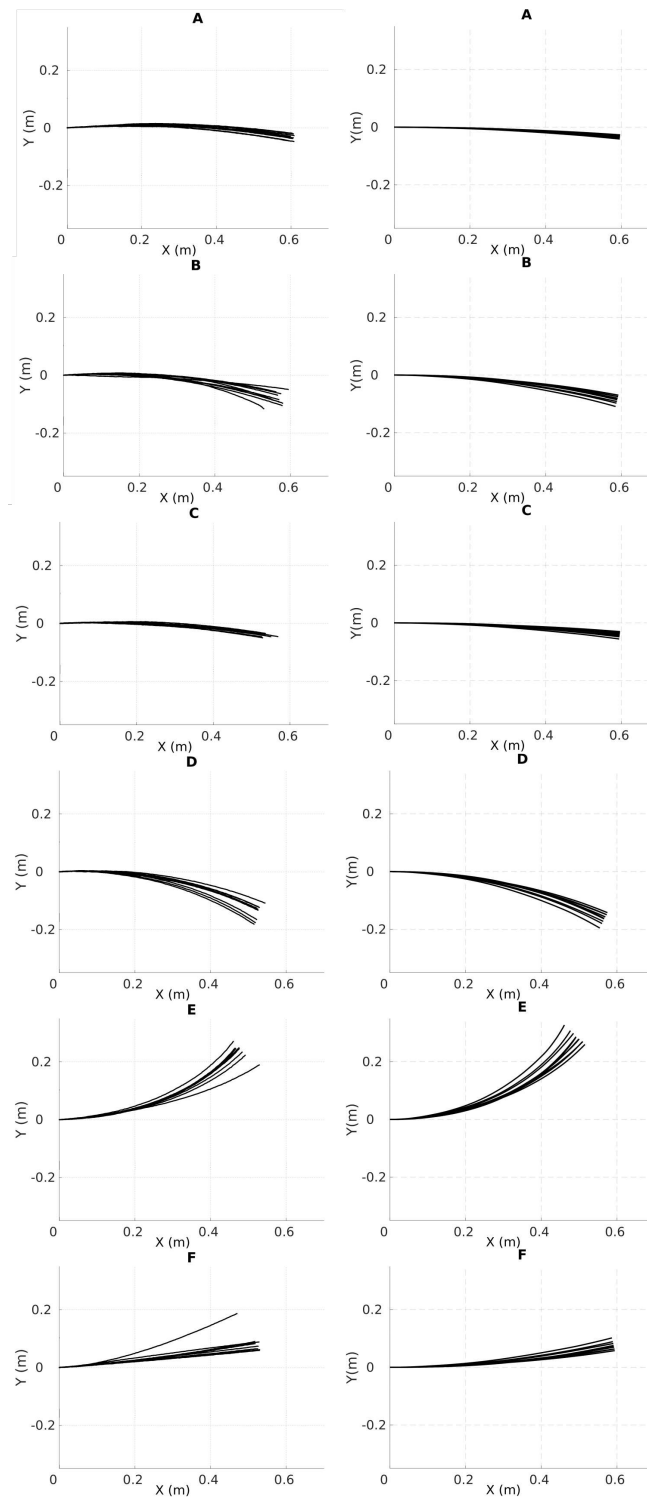


Figure 6. Trajectories of the physical (left column) robots and their respective simulated versions (right column). The letter on top of each figure represents the ID of the robot and each robot moves forwards along the x-axis for 30 seconds with a speed of 0.02 m/s and 10 iterations are shown for each robot. X(m) and Y(m) represent the coordinates of the robot.

294 (NPS and DPS) from a macro perspective. Second, through experiments in simulations and hardware, the
 295 performance of the swarm from a micro perspective is considered.

Table 2. LHA correlation values for NPS. Values in bold represent a big difference between heterogeneous and homogeneous

Parameter - output	Heterogeneous	Homogeneous
<i>Swarm size - total items collected</i>	0.95	0.97
<i>Swarm size - explore ratio</i>	-0.03	0.8
<i>Swarm size - collection ratio</i>	0.31	-0.94
<i>d - total items collected</i>	-0.87	-0.94
<i>d - explore ratio</i>	0.17	-0.99
<i>d - collection ratio</i>	-0.91	0.97

296 4.1 A study from a macro perspective using emulation

297 This section presents the results from the emulation for each strategy NPS and DPS.

298 4.1.1 Non-partitioning strategy

299 The first strategy to be described is the Non-partitioning Strategy (NPS) where the robots take the items
300 directly from the items source and transport them to the home area. P_i is the same as d and it does not
301 change across the simulation.

302 For the NPS emulation two parameters are considered: *swarm size* and d . The *swarm size* is the number of
303 robots that comprise the swarm where the range is between 2 to 14 robots. Heterogeneous and homogeneous
304 errors are considered for the emulation, in a range for d from 0.5 to 2.0 m. The width of the arena does not
305 change, and the home area and items source are against the arena walls.

306 *Latin-hypercube analysis.*

307 Results from the latin-hypercube analysis are illustrated in Figure 7, summarized in Table 2 and key
308 scatter plots are shown in Figure 8.

309 Results for heterogeneous error show that the *swarm size* term is only positively highly correlated
310 (absolute correlation coefficient greater than 0.7) with the *total items collected* output (Figure 7). This
311 means that as the *swarm size* increases, the *total items collected* output increases, and this is because there
312 are more robots collecting items in the arena.

313 The parameter d is negatively highly correlated with the *total items collected* and *collection ratio*. As d
314 decreases, it takes less time to transport the items from the items source to the home area. In addition, since
315 the distance, the robots are travelling is smaller the probability of finding items increases.

316 Results for homogeneous error show that the *swarm size* term is not only highly correlated with the *total*
317 *items collected* output but also with both the *explore ratio* and *collection ratio* (Figure 7, right). As the
318 *swarm size* increases, time spent in the *explore* state increases because robots spend more time avoiding
319 each other. As a consequence robots travel more when transporting items. Therefore, the error increases
320 and the probability of finding items decreases and is reflected in the *collection ratio* output which it has a
321 highly negative correlation with the *swarm size* term.

322 In a similar way, d is not only highly correlated with *total items collected* and *collection ratio* but also
323 with the *explore ratio* for both error types. As d increases, the probability of finding items decreases,
324 therefore, robots spend more time exploring than transporting items because they are getting lost more
325 often.

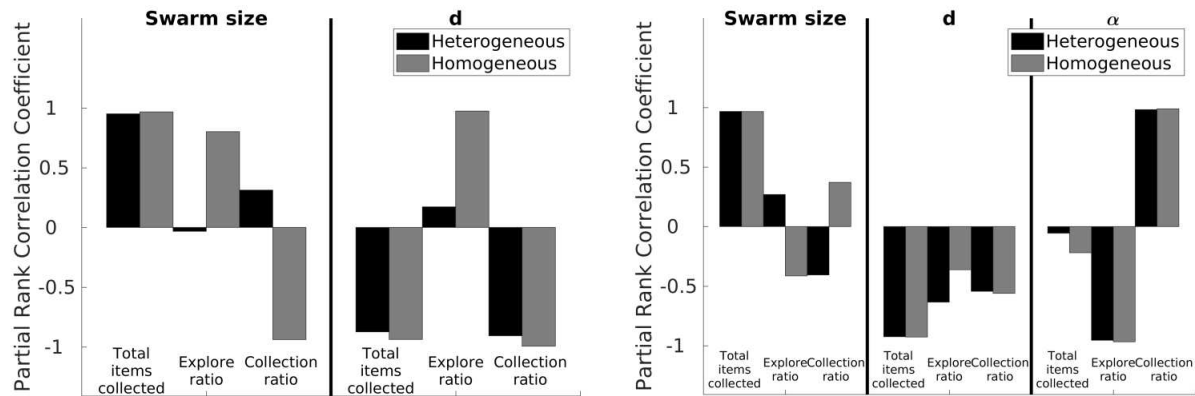


Figure 7. LHA for NPS (left) and DPS (right) with homogeneous and heterogeneous error. Left: Correlations change for each error type except for the *total items collected* for both parameters (*swarm size* and *d*) and *collection ratio* for *d* parameter. Right: results are similar between heterogeneous and homogeneous error except for correlation between the *swarm size* term and the *explore ratio* and *collection ratio*

326 Even though robots with homogeneous and heterogeneous errors are performing the same strategy, NPS,
 327 with the same settings, results from LHA differ. The main reason for this difference is that the individual
 328 errors in each robot, for the heterogeneous error, affect the performance of the swarm performance as a
 329 whole in different ways and this is explained below.

330 *Total items collected* is very similar for homogeneous and heterogeneous error as shown in Figure 8.
 331 However, as for the *explore ratio* the correlation coefficient is very different from each other, -0.03
 332 for heterogeneous error and 0.8 for homogeneous error. With the heterogeneous error model, there are
 333 fluctuations across the *swarm size* term space due to the individual errors considered. For example, between
 334 *swarm size* 2 and 3 robots (robots A, B and C), the *explore ratio* ranges from 0.4 to 0.8. This is because
 335 this group of robots are characterized by their small error compared to other robots in the swarm, as shown
 336 in the error models in the previous section (Figure 6). Therefore, these robots spend more time transporting
 337 items than exploring. However, this does not occur with the homogeneous error, as shown in Figure 8,
 338 where the *explore ratio* increases steadily without any oscillations present.

339 The *collection ratio* output for heterogeneous error also has fluctuations, as shown in Figure 8. In the
 340 regions where high error robots are introduced the *collection ratio* drops and increases again when low
 341 error robots are introduced (i.e. the range between 5 and 7 robots). However, with homogeneous error the
 342 *collection ratio* decreases steadily with no fluctuations as the swarm density increases and the robots spend
 343 more time avoiding each other. This is due to there being no high-error robots that are introduced at any
 344 point because all the robots have the same error.

345 As shown, results are different from LHA between heterogeneous and homogeneous errors for the
 346 *explore ratio* and the *collection ratios*. This is because fluctuations introduced between high and low
 347 error robots, affect the correlation coefficients. Complementary LHA scatter plots can be found on-line
 348 <https://www.york.ac.uk/robot-lab/taskpartitioning/>.

349 *Non-dominated Sorting Genetic Algorithm II.*

350 NSGA-II was used to find the Pareto front for the best values for *swarm size* and *d* parameters in order
 351 to maximize *total items collected* and *collection ratio*, and minimize *explore ratio* as shown in Figure 9.

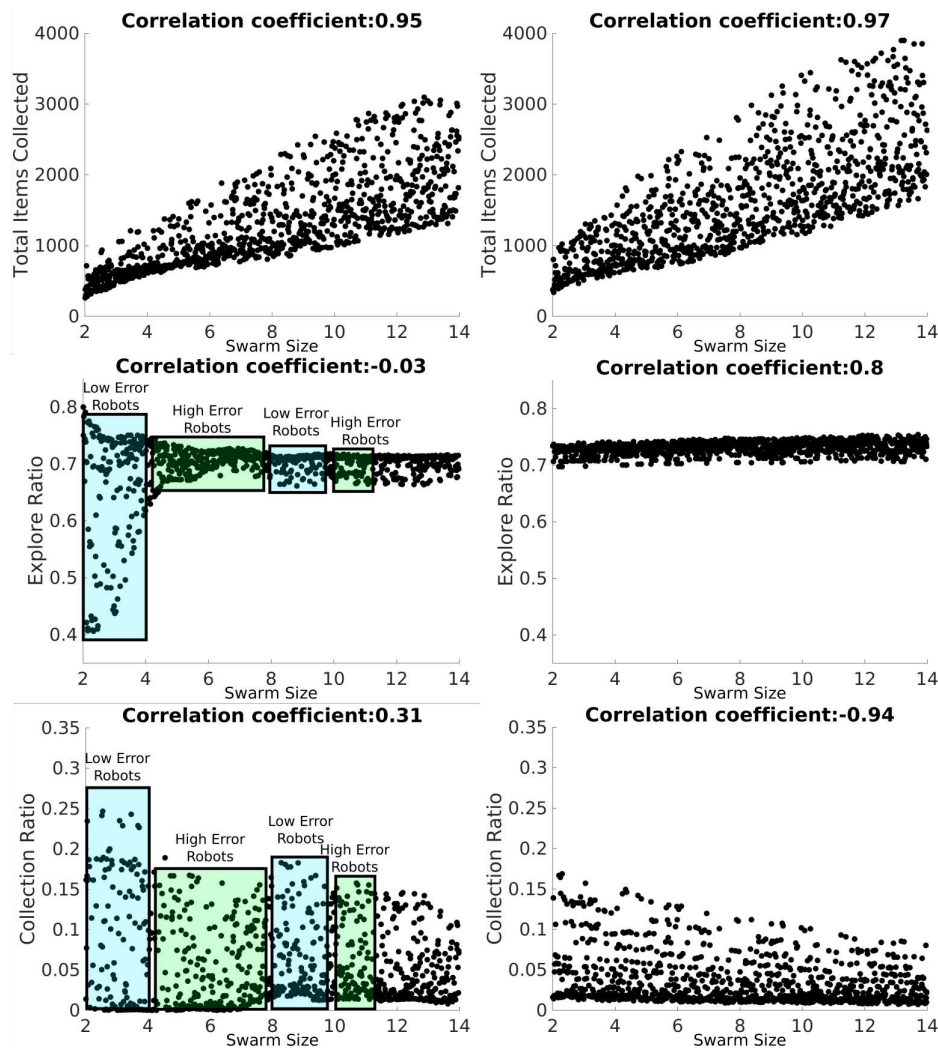


Figure 8. LHA for the *swarm size* term for NPS with heterogeneous (left column) and homogeneous (right column) errors and for the *total items collected* (top row), *explore ratio* (middle row) and *collection ratio* (bottom row) outputs. The blue rectangles represent the proportion of low-error robots and the green box represents the proportion of high-error robots. Robot individual error models introduce fluctuations with heterogeneous error as shown with the *explore ratio* and *collection ratio*. However, this doesn't happen with homogeneous error because all the robots share the same error parameters.

352 The pareto front is discontinuous for the heterogeneous error (left column), as the emulation captures the
 353 heterogeneity in the errors. For instance, for small swarm sizes, from 2 to 3 robots (robots A, B and C), the
 354 error is small, which means that the robots can travel longer distances. This minimizes the *explore ratio*
 355 and they spend the least amount of time avoiding each other since the swarm density is minor. Robots
 356 between *swarm size* 2 and 4 with d of 0.5 maximize the collection ratio. This is because the robots are
 357 travelling a shorter distance between home and source, therefore error accumulation is small. Between 5
 358 and 7 (robots E, F and G), robots are characterized by their high degree of error. Therefore, these robots
 359 behave more like obstacles and they impede item collection. As a consequence, the swarm is affected
 360 negatively. Lastly, between 8 and 14 (robots H, I, J, K, L, M and N), robots with low-error are again
 361 introduced which contribute to a high *total items collected* output. Further work could potentially use this
 362 discontinuous pareto front in order to identify the threshold of the number of high error robots that when

Table 3. LHA correlation values for DPS. Values in bold represent a big difference between heterogeneous and homogeneous

Parameter - output	Heterogeneous	Homogeneous
<i>Swarm size - total items collected</i>	0.9	0.97
<i>Swarm size - explore ratio</i>	0.42	-0.41
<i>Swarm size - collection ratio</i>	-0.29	0.37
<i>d - total items collected</i>	-0.76	-0.93
<i>d - explore ratio</i>	-0.73	-0.36
<i>d - collection ratio</i>	-0.42	-0.56
<i>α - total items collected</i>	0.07	-0.22
<i>α - explore ratio</i>	-0.97	-0.97
<i>α - collection ratio</i>	0.99	0.99

363 overcome the swarm throughput is affected negatively. This would help to measure the degrees of the
364 robustness of the task.

365 The pareto front is continuous for the homogeneous error (right column). As all the robots contribute
366 in the same way. If the *total items collected* was to be maximized the *swarm size* needs to be increased.
367 However, if the *collection ratio* is maximized and the *explore ratio* minimized the size of the *swarm size*
368 needs to be decreased. In order to optimize the three outputs it is necessary to have the smallest *d*.

369 *Summary.*

370 From this analysis, it can be concluded that it is important to consider heterogeneous and homogeneous
371 errors, as the results can be misleading if they are considered separately because of the different correlation
372 values and this validates the first hypothesis. In addition, the use of different statistical techniques helps
373 to provide a better, more in-depth, understanding of the system. NSGA-II exploits the heterogeneity and
374 finds new solutions that maximize and minimize outputs and strategies described in the following section
375 to decrease this effect by dividing the task into multiple components.

376 4.1.2 Dynamic partitioning strategy

377 The Dynamic Partitioning Strategy (DPS) is a strategy where robots change their individual partition
378 length (P_i) according to a penalty and reward mechanism. P_i changes with the parameter α .

379 α regulates the amount of penalty and reward to P_i . As α increases the robot i gets penalized and not
380 rewarded, and as α decreases the robot is rewarded more than being penalized. The interval used for the
381 experiments shown in this section is [0,1]. Initial P_i is randomly selected from a uniform distribution from
382 the same interval as d .

383 *Latin-hypercube analysis.*

384 Results from the latin-hypercube analysis are summarized in Table 3 and key scatter plots are shown in
385 Figure 10.

386 Results between heterogeneous and homogeneous error are very similar for all the correlations, except
387 for the correlation between the *swarm size* term and the *explore ratio* and the *collection ratio*, as shown in
388 Figure 7. This discrepancy is produced by the heterogeneity within the individual errors. However, the
389 correlation is kept low with the absolute correlation coefficient being lower than 0.7.

390 The *total items collected* output is, once again, mainly correlated to the *swarm size* and *d* parameters for
391 the reasons explained in the previous section. However, α has a negative correlation with *explore ratio* and

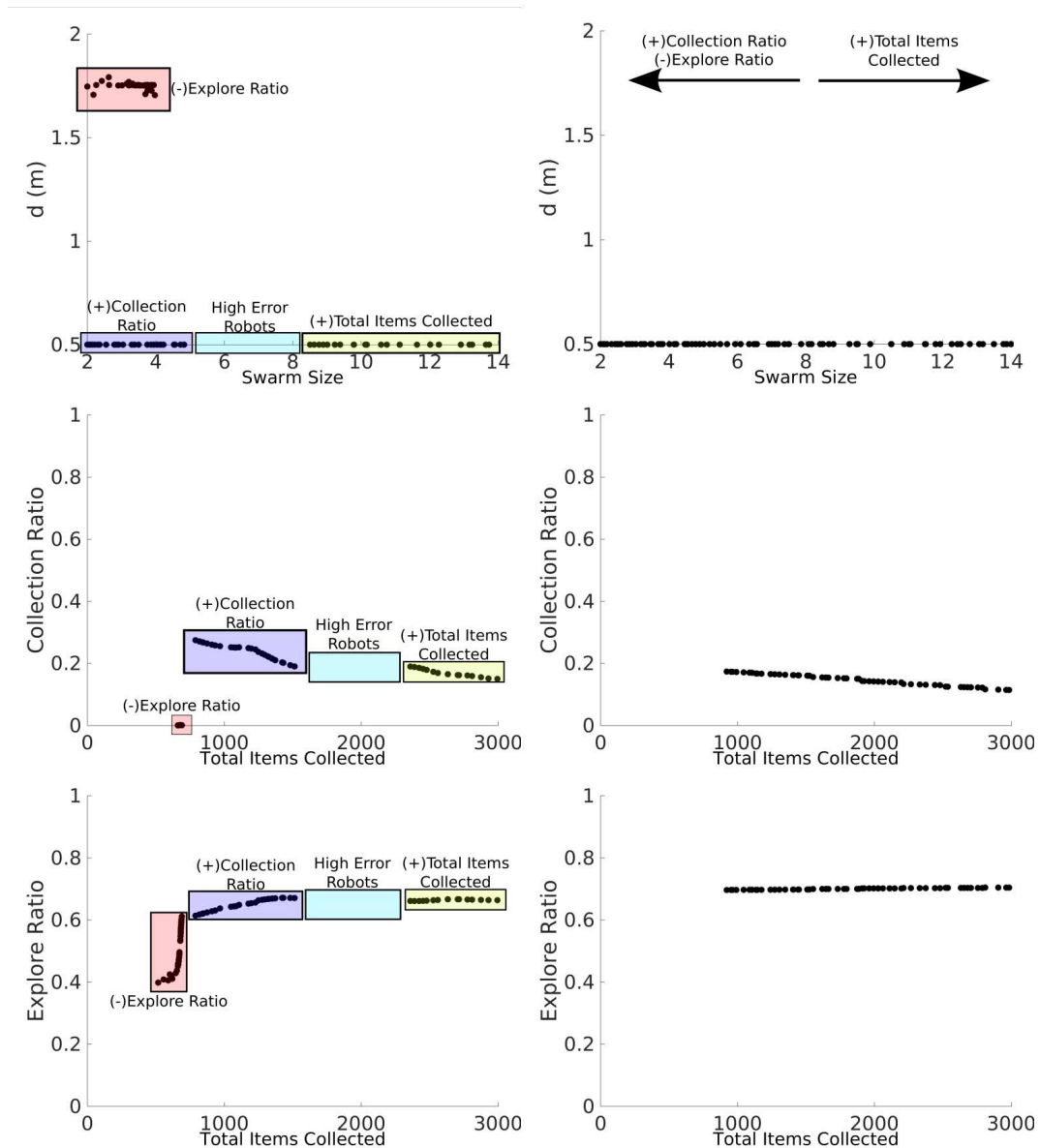


Figure 9. NSGA-II results for NPS with heterogeneous (left column) and homogeneous (right column) error. The two-dimensional inputs are *swarm size-d* (top row). Two-dimensional outputs are shown as *total items collected-collection ratio* (middle row) and *total items collected-explore ratio* (bottom row). With both the heterogeneous error and the homogeneous error, the *total items collected* are maximized with high *swarm sizes* (yellow box). With both the heterogeneous error and the homogeneous error, the *collection ratio* is maximized with low *swarm sizes* (blue rectangles). The *explore ratio* is maximized with low *swarm sizes* and high *d* for heterogeneous error and low *swarm sizes* and low *d* with homogeneous error. The latter result demonstrates that since this proportion of robots experiences a low error, it is better for these robots to travel a longer *d* as the robot density is lower the robots spend less time avoiding each other. The Pareto front is discontinuous for the heterogeneous error which means that these robots with high error harm more the item collection than contribute to it.

392 a positive correlation with *collection ratio*, This means that as α increases the robots are exploring less and
 393 finding items more often because all the robots are adjusting their P_i . In other words, as α increases robot
 394 are travelling shorter distances which allows for less dead-reckoning error accumulation.

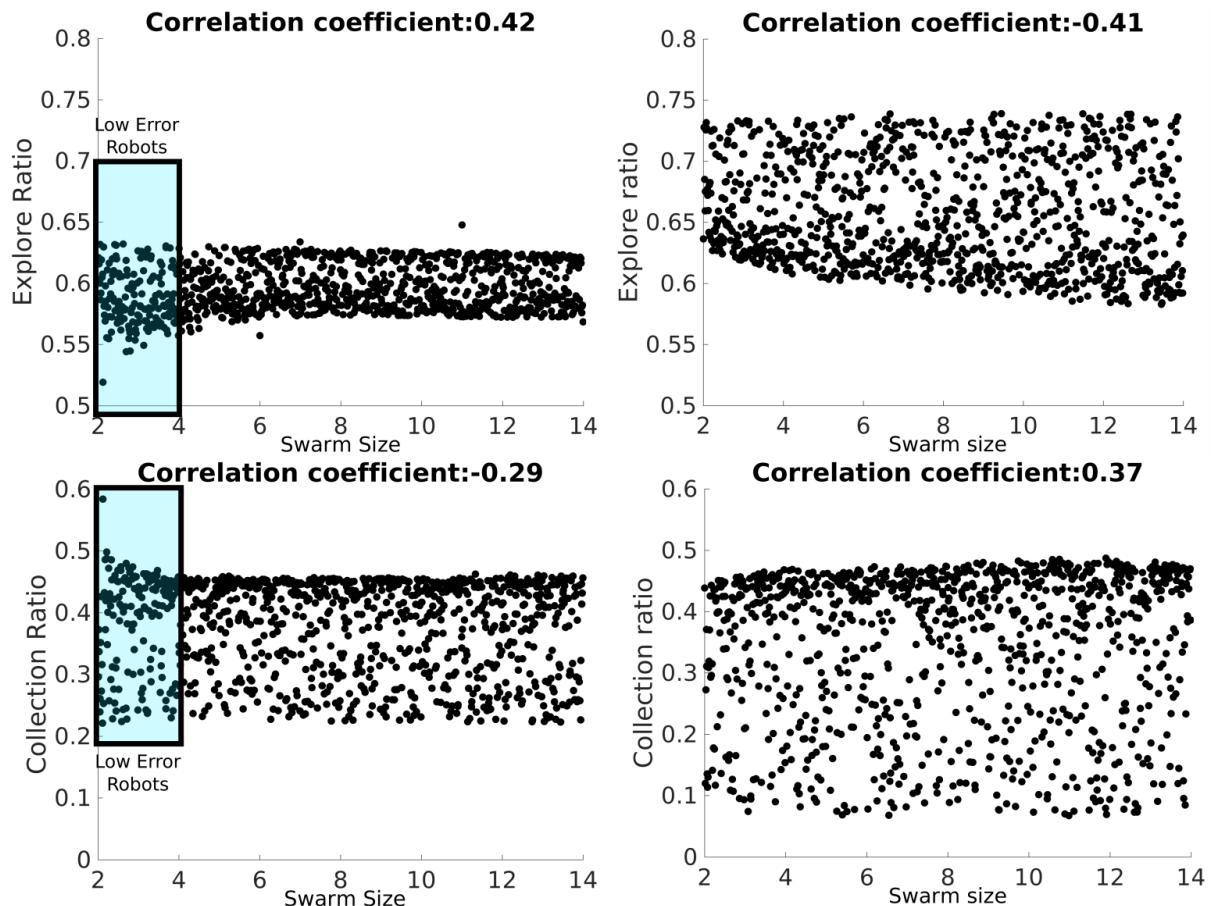


Figure 10. LHA for the d parameter for DPS with heterogeneous (left column) and homogeneous (right column) errors and the explore ratio (top row) and collection ratio (bottom row) outputs. The blue rectangle represents the proportion of low-error robots.

Fluctuations introduced by low-error robots affect the estimation of the correlation coefficient.

395 The discrepancy mentioned before for the *swarm size* term is shown in Figure 10. For the heterogeneous
 396 error, the *collection ratio* decreases and the *explore ratio* increases, whereas as for the rest of the swarm
 397 this correlation pattern can not be seen. This could be due to high error robots compensating for the
 398 heterogeneous error. Whereas for the homogeneous error, the *explore ratio* and the *collection ratio*
 399 decreases and increases steadily respectively for the homogeneous error. As the swarm size increases, there
 400 is a higher supply of robots which allows for the creation of a chain successfully between the home area
 401 and the source.

402 *Non-dominated Sorting Genetic Algorithm II.*

403 Large swarm sizes provide the best item collection and α regulates the *explore ratio* and ratio *collection*
 404 where as α increases, the *collection ratio* increases and the *explore ratio* decreases, see (Figure 11) for this
 405 effect with the heterogeneous error. In a similar way, NSGA-II for the heterogeneous error exploits the
 406 small swarm size in order to achieve the best *explore ratio* and *collection ratio*.

407 The Pareto front for DPS is smooth and discontinuities, or gaps, are smaller than NPS. This is because
 408 DPS has better regulation over individual errors homogenising the swarm. Each robot i is learning its own
 409 P_i for the given α (see next section for more details). This can be also seen for the α and *swarm size*
 410 inputs where there are only two batches of solutions, the ones closer to swarm size of 2 robots and the ones

Table 4. Parameter values for micro perspective experiments

Parameter	Value
Experiments length	5 hours
Swarm size	6
d	1 m
P	0.5 m
α	0.5

411 closer to the swarm size 14. This means that fluctuations by high and low error robots between the intervals
412 4 and 10 are ignored and have no significant contributions to the results.

413 Results for the homogeneous error show that a larger swarm provides the best item collection and as
414 α increases the *collection ratio* increases and the *explore ratio* decreases. The Pareto front for the set of
415 solutions is continuous and smooth.

416 *Summary*

417 α helps to regulate the *collection ratio* in order to optimize the *explore ratio*. The *total items collected*
418 output is provided mainly by the *swarm size* and d . In this way, it is possible to find a set of parameters that
419 provides a good trade-off between the *explore ratio* and the throughput of items according to the needs
420 of the user. Finally, fluctuations caused by individual error models are reduced and a continuous Pareto
421 front for the output is generated for the heterogeneous error, due to individual convergence of P_i for each
422 robot and, since the robots are travelling shorter distances, the error is lower and more uniform. More
423 information about P_i convergence can be found in the experiments from simulation and hardware in the
424 next section.

425 4.2 A study from a micro perspective using simulator and hardware

426 In the previous section, the effect of considering individual robot errors in emulation with each strategy
427 was explored. An emulator was used to aid with this study and it was found that results differ for
428 heterogeneous and homogeneous error. In this section, this discrepancy is explored in detail by using
429 experiments from a micro perspective with simulations and hardware (stages 2 and 3 from the experimental
430 framework shown in Figure 2).

431 The parameter values chosen, unless stated, for the experiments in this section are shown in Table 4.

432 4.2.1 Heterogeneous and homogeneous errors

433 As discussed in Section 3.4 when modelling the error for each robot, it was found that the error varies
434 between robots. This section describes error diversity and non-diversity in simulations (Figure 12).

435 In the first set of experiments, the robots are performing DPS where P_i is the same as d . Convergence of
436 P_i is shown in Figure 13. All the robots converge to a single similar P with homogeneous error. However,
437 robots with heterogeneous error converge to different P_i for each robot i .

438 The impact of different errors is reflected in the individual performance. In experiments shown in
439 Figure 14, all robots start with a P_i of 0.4 m and α of 0.5 with homogeneous and heterogeneous error
440 models for DPS. The robots with homogeneous error have a similar individual performance as each other.
441 However, robots with heterogeneous error have different individual performance according to each robot.

442 Robot specialization emerges from DPS in the sense that every robot learns its own P_i according to
443 their inherent degree of error as shown in Figure 13. This is consistent with the results presented in

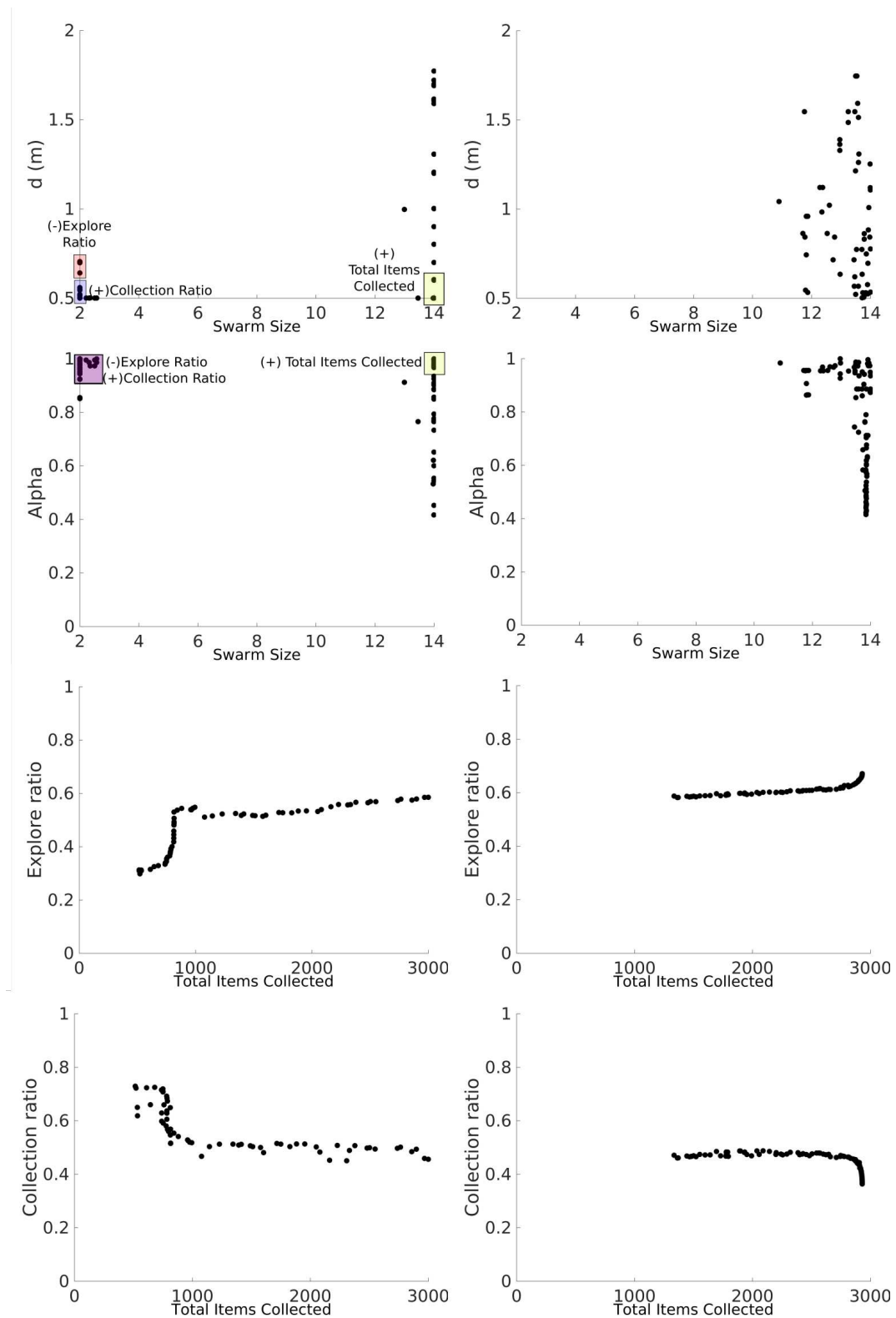


Figure 11. NSGA-II results for DPS with heterogeneous (left column) and homogeneous (right column) error. Two-dimensional inputs are shown as *swarm size-d* (first row) and *swarm size-alpha* (second row). Two-dimensional outputs are shown as *total items collected-explore ratio* (third row) and *total items collected-collection ratio* (fourth row).

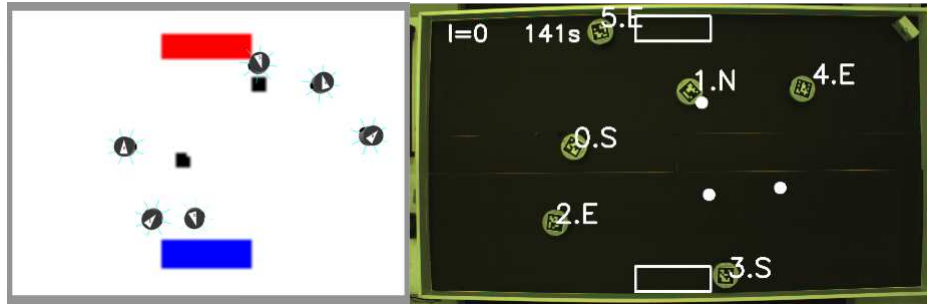


Figure 12. Screenshot of one of the experiments in simulation (left) and hardware (right).

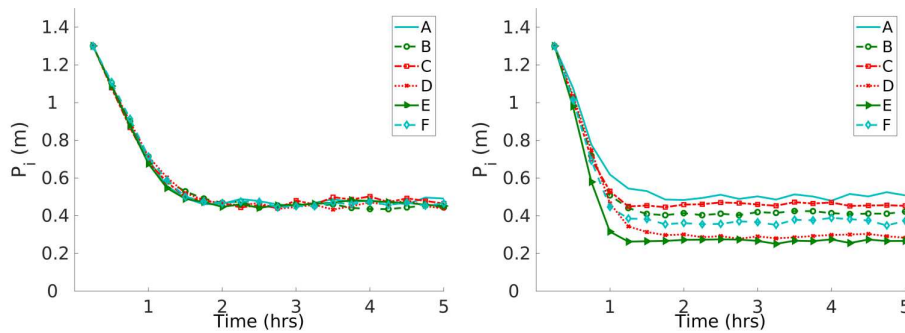


Figure 13. Convergence to a solution with homogeneous (left) and heterogeneous (right) errors with DPS. Robots with homogeneous error converge to a single P_i and robots with heterogeneous error converge to different P_i .

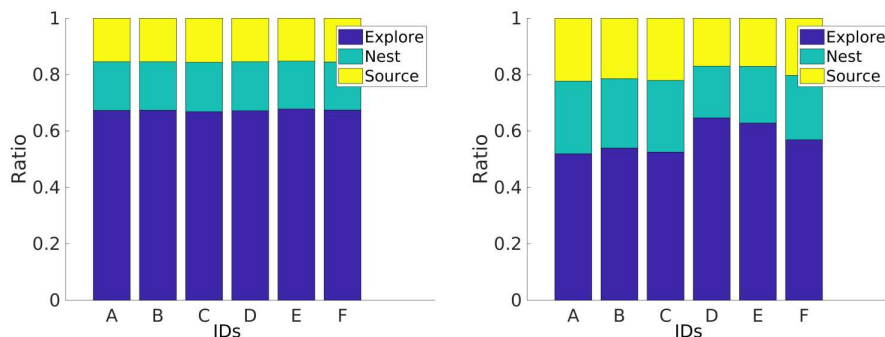


Figure 14. Individual performance with homogeneous (left) and heterogeneous (right) error models. Robots with homogeneous have similar individual performance and robots with heterogeneous error have different individual performance.

444 Figure 14 where the P_i for each robot is correlated with the amount of time that a robot spends exploring
 445 the environment. For example, robot A spends the least amount of time in the explore state and, hereby,
 446 has the greater P_i with a value close to 0.5 m. Robots D and E spend the greatest amount of time in the
 447 explore state and these robots travel the small P_i with a value close to 0.3 m. In addition, this consistent
 448 with the trajectories shown in Figure 6.

449 For instance, Figure 13 shows that robots D, E and F learn the smallest P_i and this is related to their
 450 individual performance where these robots spend more time in the *explore* state than the rest of the robots
 451 as shown in Figure 14. In addition, Figure 6 shows that these same robots have bigger error drifts.

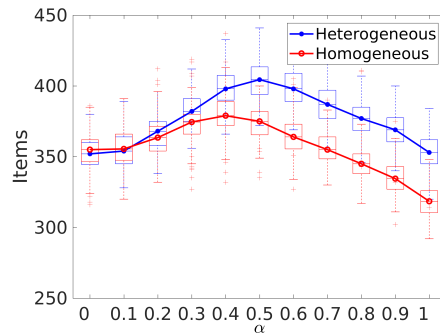


Figure 15. Item collection with homogeneous and heterogeneous errors with different values for α . The peak for item collection is 0.4 for homogeneous and 0.5 for heterogeneous error model.

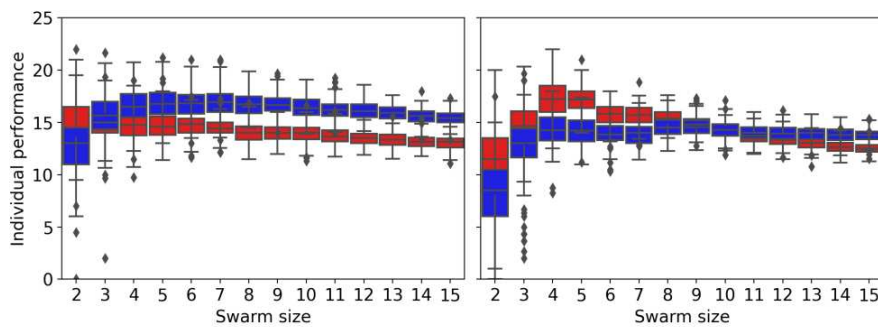


Figure 16. Individual performance for different swarm sizes for the homogeneous error (left) and heterogeneous error (right) for NPS in red and DPS in blue. The individual performance decreases after the first robot for NPS and after 7 robots for DPS with the homogeneous error. The individual performance peaks at 4 robots for NPS and at 8 robots with DPS for the heterogeneous error.

452 It is important to bear in mind the difference in performance between heterogeneous and homogeneous
 453 errors, in order to select the appropriate value of α for the swarm to maximize the item collection as shown
 454 in Figure 15. In the case of a homogeneous error model, α should be 0.4 in order to have the best item
 455 collection. As with the heterogeneous error model, that value would be 0.5.

456 Figure 16 shows the individual performance for different swarm sizes, strategies (NPS and DPS) and
 457 error types (homogeneous and heterogeneous). The individual performance is estimated by dividing the
 458 *total items collected* by the *swarm size*. For the *homogeneous error* the individual performance starts to
 459 drop after swarm size 2 for NPS and after swarm size of 7 for DPS. The results differ with the *homogeneous*
 460 *error* and this is due to the mix of low and high error robots in the swarm as mentioned in the previous
 461 section. For example, NPS peaks at 4 robots because the first 4 robots experience low error and after this,
 462 the robots introduced experience high error as shown in Figure 8. This is not the case for DPS where the
 463 distribution resembles more like a bimodal distribution with two peaks at 4 and 9, each peak at the location
 464 where robots with low error are introduced. More results regarding the robot swarm flexibility against a
 465 changing d , change of error models and change of *swarm size* can be found in the supplementary material
 466 <https://www.york.ac.uk/robot-lab/taskpartitioning/>.

467 Indirect item transference is the method chosen for most of the experiments shown in this section because
 468 it has the highest item collection with the lowest convergence speed. This is an important aspect to consider
 469 due to the battery life of the physical robot only lasting 30 minutes.

470 To conclude, the above results demonstrate that each robot exhibits different performance and this
471 validates the second hypothesis. It is important to consider individual models for each robot when working
472 with swarm robotics. By doing so, it is possible to reduce the reality gap and be able to select the best α
473 that maximizes the item collection and, in addition, potentially select the best P_i and α for each robot to
474 improve the item collection. In Section 4.2.3 experiments with physical robots are shown and how they
475 compare with simulations.

476 4.2.2 Social entropy

477 Results from NSGA-II in emulation for NPS showed a discontinuous Pareto front for the heterogeneous
478 error and a continuous Pareto front for the homogeneous error (Figure 9). In this section, these results are
479 explored in more detail to understand the reason of this discrepancy, using an approach based on a social
480 entropy metric. This metric measures the error diversity across the swarm and is described in Section 3.1.

481 Decision trees and nearest neighbour classifiers are used to categorize robots into two groups according
482 to their individual performance. Principal Component Analysis (Wold et al., 1987) is used to convert
483 the variables from the individual performance data to their principal components. This is done in order
484 to transform possibly correlated variables into a set of values of linearly uncorrelated variables. The
485 requirement for classification is that the accuracy should be higher than 90%. Figure 17 illustrates the
486 social entropy for each swarm size in the interval [2,14] for each strategy. The social entropy is overlapped
487 with the results from the NSGA-II reported in Section 4.1.

488 The social entropy for NPS illustrates that the Pareto front breaks when the social entropy highest peak
489 is reached. For a swarm interval between 2 and 4 robots, the social entropy is low. This means that the
490 swarm is homogeneous and composed of robots with low degree of error. This is the reason for the cluster
491 of points in d of 1.75 m. However, after the first robot with high degree of error is introduced (robot D or
492 fourth robot), the social entropy increases. At this stage, the swarm is homogeneous enough to have the
493 Pareto front continuous. However, the discontinuity in the Pareto front appears when the second robot with
494 high degree of error is introduced (robot E or fifth robot). The social entropy reaches its highest peak point
495 at this point. The social entropy starts to decrease after the sixth robot (robot F) is introduced. The social
496 entropy decreases steadily because more robots with low degree of error are added to the swarm and this
497 decreases the diversity. The Pareto front reappears after the ninth robot is introduced. After this point the
498 Pareto front is similar to the homogeneous error Pareto front (Figure 9).

499 Overall, the social entropy across the swarm for DPS is similar to with NPS. The social entropy peak
500 again is located after the fifth robot and the Pareto front becomes discontinuous after this point. However,
501 in contrast to NPS the social entropy remains high (greater than 0.8). This prevents the Pareto front from
502 being continuous until the social entropy drops below 0.8. DPS creates a clear distinction between high
503 and low degree of error and because of this, the social entropy increases. This might be because task
504 partitioning is sensible for high noisy degree of error robots.

505 As noted the social entropy changes with each strategy and the reason is that each strategy provides a
506 different performance for each robot. For NPS, the highest degree of error robots classified are 2, robots
507 D and G. DPS, for this scenario, the robots experience the least amount of time in the explore state. The
508 number of high-degree error robots increases to 4 (D, E, G and J).

509 In the case of DPS, robots D, E, G and J experience the highest degree of error. Therefore, these robots
510 can not be further optimized. This is consistent with the results shown in Figure 17 where the P_i for robots
511 D and E with a d of 0.4 m does not increase indefinitely as with robots A, B, C and F. This means that

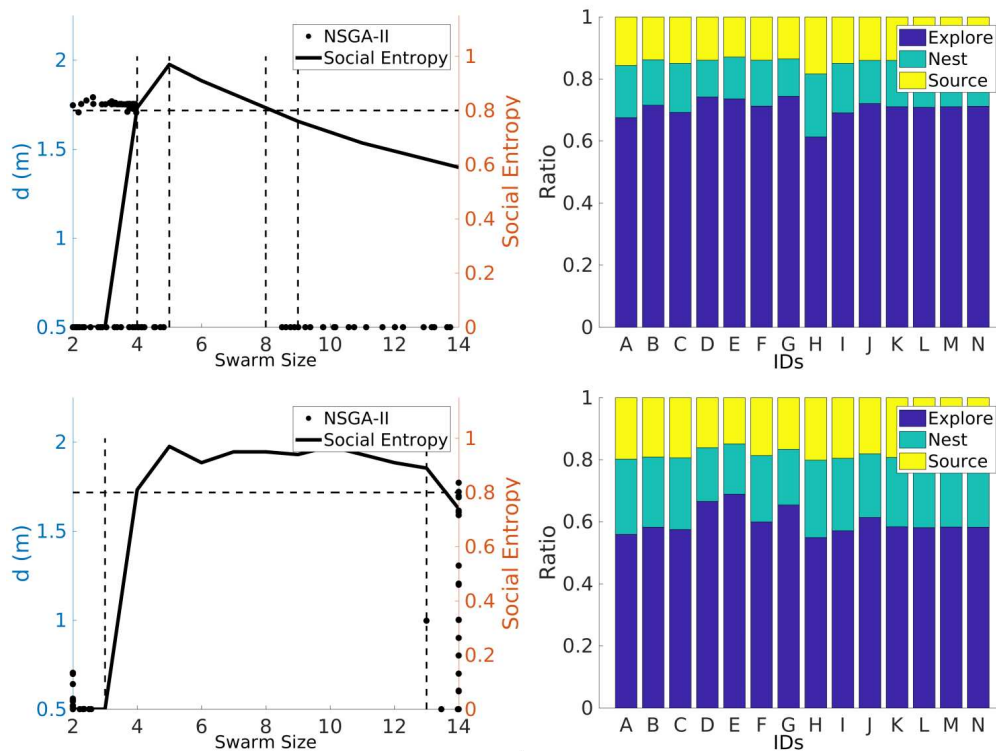


Figure 17. Social entropy and individual performance (left and right column) for each strategy: NPS and DPS (first and second rows respectively). Social entropy changes for each strategy due to the number of robots classified as high error.

512 robots have not reached the optimal distance that decreases the error. In other words, robots D, E, G, and J
 513 experience such a high degree of error, that the partitioning strategies are unable to provide a suitable P_i
 514 for the robots to increase their performance.

515 4.2.3 Experiments with physical robots

516 In this section, the experiments studied in the previous section are explored with physical robots. Three
 517 different arena sizes are used for the experiments using 6 robots where individual performance and
 518 convergence are explored. A single replicate for each arena is shown in this section. It is important to
 519 mention definite conclusions cannot be made from the results shown in this section due to this low sample
 520 size, however the results are promising. A screenshot from an experiment is shown in Figure 12.

521 Results for robots performing NPS are shown in Figure 18. Regardless of the arena size, the robots
 522 experience different individual performance which means that they are not identical to each other. In
 523 addition, robots spend most of their time exploring instead of retrieving items in the home area. Finally,
 524 robots spend less time in the *go to source* than *go to nest* state. This is because since robots are travelling a
 525 longer P_i they experience a greater drift from the original position, causing the target to be beyond the
 526 walls of the small arena as previously seen. This effect decreases as the arena increases because the target
 527 is within the arena boundaries.

528 Individual performance for robots with DPS can be found in Figure 19. In a similar way to experiments
 529 with NPS, robots performing DPS experience an increment in the amount of time spent in the *explore* state
 530 as the arena size increases. In contrast with NPS, robots spend more time in the *going to source* state. The

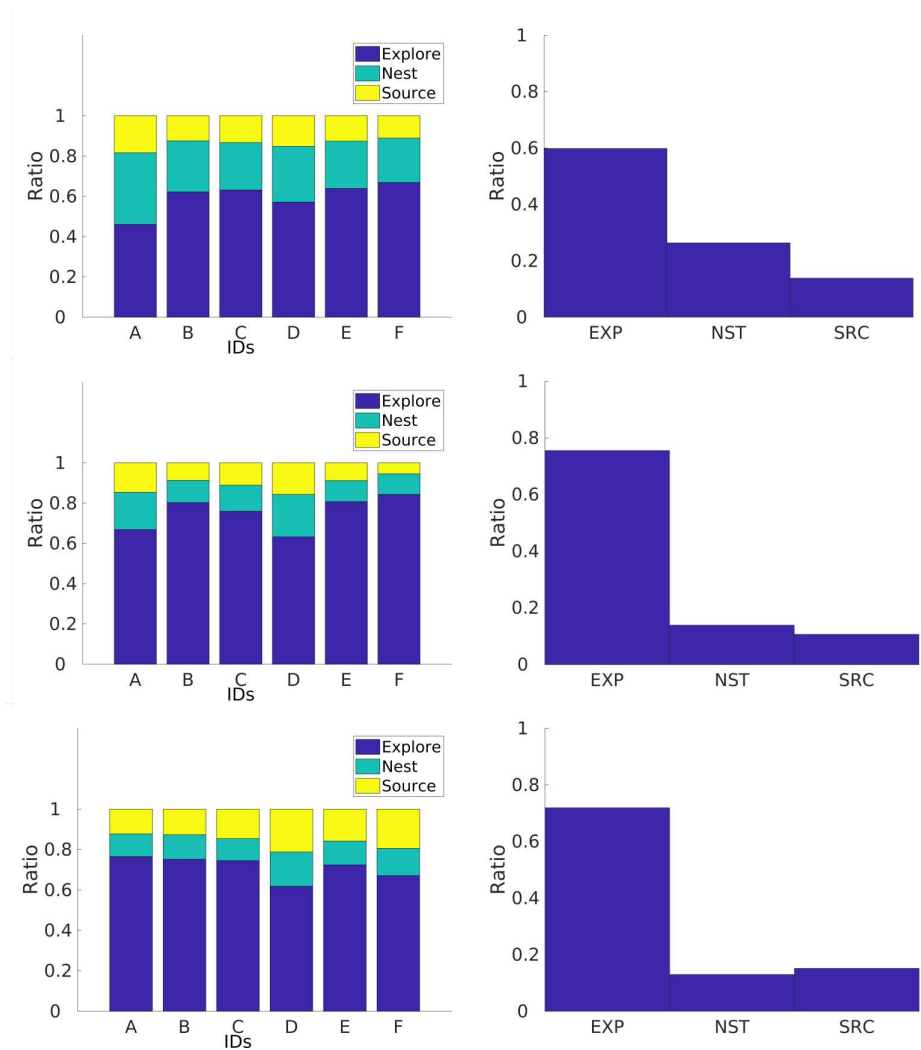


Figure 18. Individual performance for physical robots performing NPS for three different arenas with dimensions of 1.0 x 1.3 m (top row), 2.1 x 1.2 m (middle row) and 2.4 x 1.6 m (bottom two). The go to source rate is smaller than the go to nest as the arena becomes smaller.

531 reason for this, is that the robots are travelling shorter P_i which decreases the probability of the target being
 532 beyond the arena size, for the small arena, in a similar way to that in the simulations.

533 Since the battery of these robots last 30 minutes, the convergence experiments were divided into three
 534 different sets shown in Figure 21. In each set, the robots started with a different P of 1, 0.7 and 0.4 m for
 535 an arena of size 2.1 x 1.2 m. All the robots converge to a distance close to 0.4 m that changes from robot to
 536 robot. The velocity of convergence is not only related to the degree of error in the robot but also, to the
 537 speed of the robot which changes slightly for each robot. Furthermore, from the figures can be seen that
 538 there is no change in the P_i at least for the first 5 minutes of the experiments. This is the time that it takes
 539 to find the first item.

540 As for the item collection, DPS performs better than NPS as the arena size increases as shown in Figure 21
 541 because of the following issues. First, as the area of the arena decreases, the probability of finding the items
 542 source increases even though the collection rate is low. This means that partitioning is not required. Second,
 543 robots are spending more time dropping and picking up items with DPS which causes a delay in the item
 544 getting to the nest. Finally, robots spend more time avoiding each other when they are in the *go to nest*.

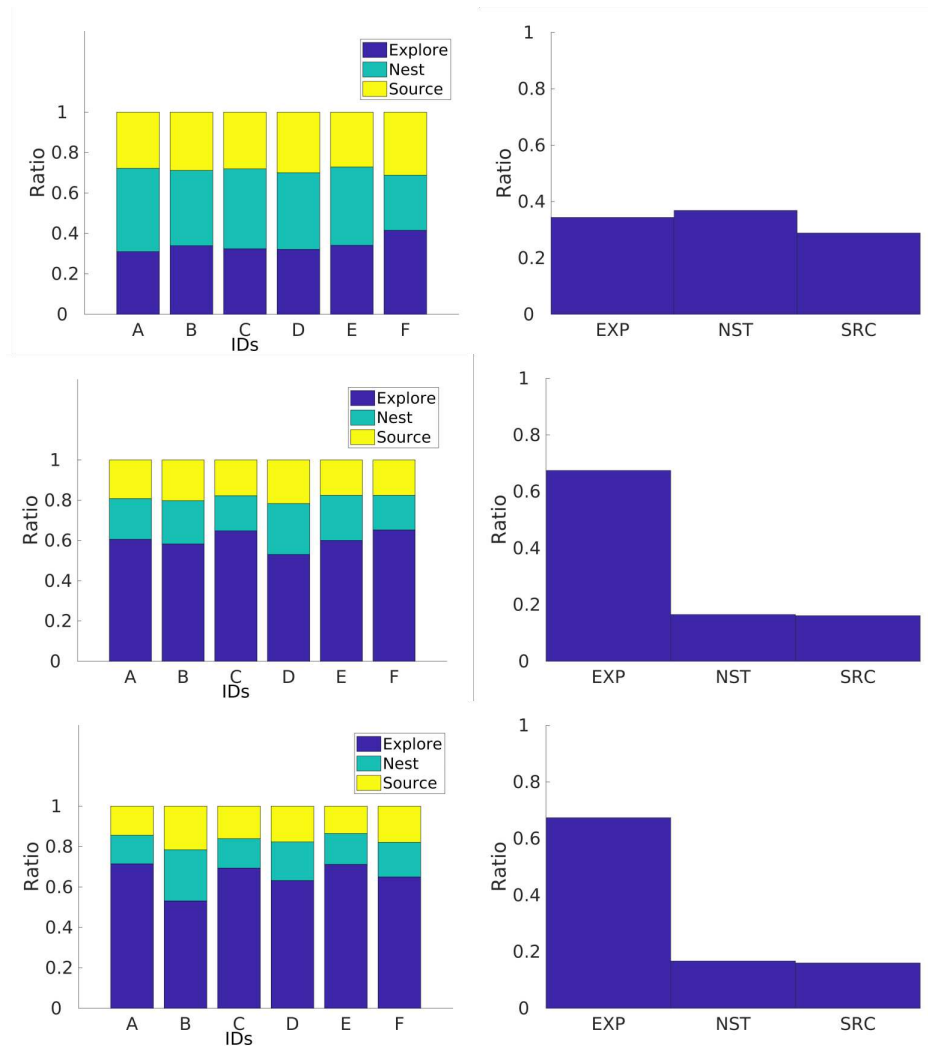


Figure 19. Individual performance for physical robots performing DPS for three different arenas with dimensions of 1.0 x 1.3 m (a), 2.1 x 1.2 m (c) and 2.4 x 1.6 m (e). Time spent in the *go to source* state is roughly similar to the *go to nest* state and time spent in the *explore* state is lower than with DPS.

545 In summary, results from experiments with physical robots are similar to results from simulation. The
 546 shape of the arena affects the performance of the *go to source* state mainly for NPS. DPS performs better
 547 than NPS as the size of the arena increases. Finally, final P_i changes according to the amount of error in
 548 the robot i . However, there are some differences due to the reality gap because the models used in the
 549 simulator do not provide enough information about the real world.

5 DISCUSSION

550 It is a common assumption that all the robots are similar and a single robot model in simulations represents
 551 the entire group of robots in a robotic swarm. Furthermore, it was assumed that all robots would have the
 552 same behaviour when performing a task. However, work in this paper has shown that this is not necessarily
 553 the case.

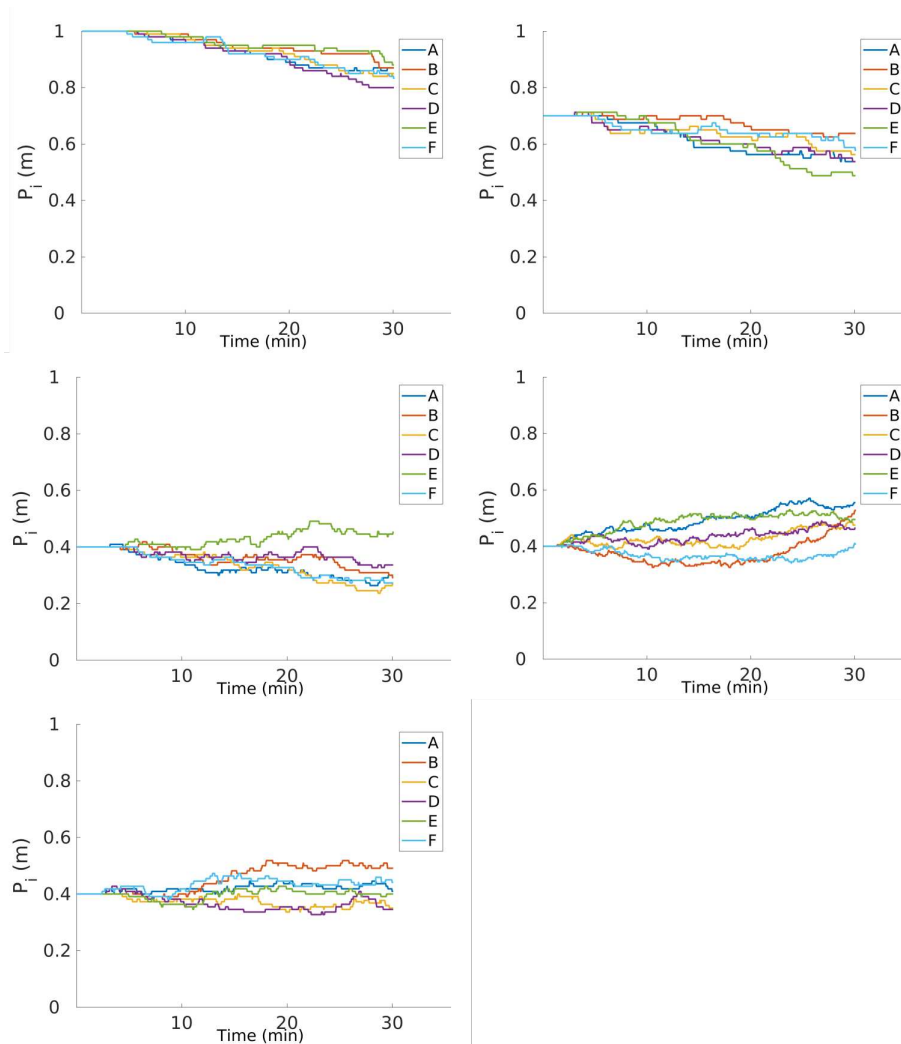


Figure 20. Convergence of P_i for different *initial partition length* (P_i^0) of 1.0 (top left), 0.7 (top right) and 0.4 m (middle left) for an arena of 2.1 x 1.1 m. Convergence for arenas 1.0 x 1.3 m (middle right) and 2.4 x 1.6 m (bottom left) for P_i^0 of 0.4 are also shown. The convergence changes from robot to robot and is due to the error and velocity of the robot.

554 At the moment of retrieving the trajectories for each robot as shown in Section 3.4 each robot experiences
 555 a different degree of error. Each robot has a different bias, small or large, of moving towards the left or
 556 right direction. The trajectories of 14 robots was modelled and recorded.

557 In the enriched analysis in Section 4.1 it was shown that two different patterns emerged from
 558 heterogeneous and homogeneous errors for each strategy. Each pattern represents different properties of the
 559 task partitioning approach. On one hand, the pattern shown with homogeneous error provides information
 560 of the interactions of each parameter with each output. On the other hand, the pattern with heterogeneous
 561 error provides information how the swarm copes with robots with high and low degree of error for each
 562 strategy.

563 The individual contribution of each robot changes according to its inherent error as shown in Section 4.2.
 564 If the homogeneous error is considered, all the robots converge to the same P_i value. However, if the
 565 heterogeneous error is considered, then each robot converges to a different P_i . P_i is able to adapt to
 566 changes in the error itself, d and swarm size.

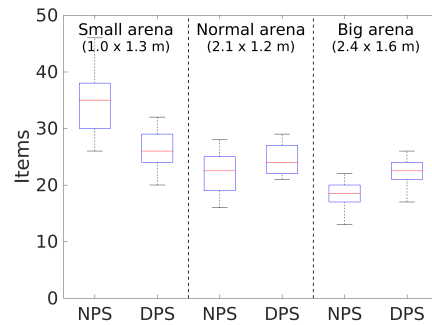


Figure 21. Item collection with physical robots and different arena sizes. DPS starts to overcome NPS as the arena size increases.

567 The social entropy measurement for each strategy bridges the results between the macro and micro
 568 perspective (Section 4.2.2). The number of high-degree error robots plays an important role at the moment
 569 of defining optimality. As the number of high degree of error robots increases, this affects the range of
 570 optimal solutions. In addition, even though the task partitioning strategies increase the performance of each
 571 robot, there are a specific number of robots that gain no benefit from task partitioning, due to their high
 572 degree of error.

573 Finally, the experiments with physical robots shown in Section 4.2.3 validate the previous results in
 574 terms that each robot performs differently and each robot converges to a different solution according to the
 575 inherent error.

576 The work presented in this paper is limited to a single task, a single robot platform and a single type of
 577 error but there is no reason this work could be applied to other domains. Here are some examples of future
 578 work:

- 579 • The work presented in this paper could be extended to other tasks such as collective object transportation
 580 and collective decision-making. In collective object transportation, there is a high reliance on the input
 581 from the sensors and the error in the sensors could be studied. In collective decision-making, error
 582 diversity could lead to false positives in decisions.
- 583 • The error diversity could be analysed for other robot platforms and this could lead to interesting results.
 584 On one hand, the error diversity is so uniform that the approach in this paper becomes insignificant, on
 585 the other hand, the error diversity could be greater and its importance becomes greater.
- 586 • In this paper it was shown that the number of high error robots changes the dynamics of the swarm
 587 behaviour. Swarm diagnosis and recovery can be implemented to address these robots with errors.

588 In conclusion, it is important to consider both, homogeneous and heterogeneous errors, in order to have a
 589 comprehensive understanding of the performance of a swarm.

6 CONCLUSION

590 Task partitioning strategies have been shown to increase performance in foraging tasks in robotic swarms
 591 with dead-reckoning noise (Pini et al., 2013, 2014; Buchanan et al., 2016). However, a common assumption
 592 is that all the robots in a swarm share the same error model. In this paper how different degrees of error
 593 affect the swarm were studied.

594 In this paper, work has shown how each robot in the swarm experiences different degrees of error
595 (heterogeneous error) and there is a single degree of error that describes all the robots (homogeneous error)
596 and results undertaking a foraging task differ when considering heterogeneous and homogeneous error.
597 Work has shown that it is important to consider both error types to have a full understanding of the system.
598 Finally, the number of high degree of error in the swarm defines optimality in the system. The degrees of
599 robustness can be measured by identifying the ranges of optimality and the reality gap is able to be reduced.
600 Work in this paper has also shown that it is possible to distinguish robots that harm the performance of the
601 swarm.

602 Further work will consider measuring the degrees of robustness in tasks other than foraging. It will be
603 important to examine how a range of different degrees of errors would affect fault detection and diagnosis.
604 Our hypothesis is that the number of false positives would increase and a threshold that differentiates robots
605 between faulty and non-faulty would be required.

AUTHOR CONTRIBUTIONS

606 The individual contribution for this article were as follows: conceptualization: E.B., K.A., A.P., J.T.,
607 A.M.T.; methodology: E.B., K.A.; software: E.B.; validation: E.B., K.A.; formal analysis: E.B., K.A.;
608 investigation: E.B., K.A., A.P., J.T., A.M.T.; visualization: E.B.; supervision: E.B., A.P., J.T., A.M.T.;
609 project administration: E.B., K.A., A.P., J.T., A.M.T.; funding acquisition: E.B.; writing—original draft:
610 E.B., K.A., A.P., J.T., A.M.T.. All authors have read and agreed to the published version of the manuscript.

FUNDING

611 EB acknowledges financial support from CONACyT. JT is part sponsored by The Royal Society.

DATA AVAILABILITY STATEMENT

612 The code to replicate the experiments shown in this paper can be found in [https://](https://github.com/edgarbuchanan/psiswarm_task_partitioning)
613 github.com/edgarbuchanan/psiswarm_task_partitioning and [https://github.](https://github.com/edgarbuchanan/psiswarm_model)
614 [com/edgarbuchanan/psiswarm_model](https://github.com/edgarbuchanan/psiswarm_model).

REFERENCES

- 615 Alden, K., Cosgrove, J., Coles, M., and Timmis, J. (2018). Using emulation to engineer and
616 understand simulations of biological systems. *IEEE/ACM Transactions on Computational Biology*
617 *and Bioinformatics*
- 618 Alden, K., Read, M., Andrews, P. S., Timmis, J., and Coles, M. (2014). Applying spartan to Understand
619 Parameter Uncertainty in Simulations. *The R Journal* 6, 1–18
- 620 Alden, K., Read, M., Timmis, J., Andrews, P. S., Veiga-Fernandes, H., and Coles, M. (2013). Spartan:
621 a comprehensive tool for understanding uncertainty in simulations of biological systems. *PLoS*
622 *computational biology* 9, e1002916
- 623 Arkin, R. (1987). Motor schema based navigation for a mobile robot: An approach to programming by
624 behavior. *Proceedings. 1987 IEEE International Conference on Robotics and Automation* 4, 264–271.
625 doi:10.1109/ROBOT.1987.1088037
- 626 Balch, T. (1998). *Behavioral diversity in learning robot teams*. Ph.D. thesis

- 627 Balch, T. (2005). Communication, diversity and learning: Cornerstones of swarm behavior. *International*
628 *Workshop on Swarm Robotics* , 21–30doi:10.1007/b105069
- 629 Bjercknes, J. D. and Winfield, A. F. T. (2012). On fault tolerance and scalability of swarm robotic systems.
630 *Springer Tracts in Advanced Robotics* 83 STAR, 431–444. doi:10.1007/978-3-642-32723-0_31
- 631 Bongard, J. C. (2000). The Legion System: A Novel Approach to Evolving Heterogeneity for Collective
632 Problem Solving. *European Conference on Genetic Programming* , 16–28
- 633 Brutschy, A., Pini, G., Pinciroli, C., Birattari, M., and Dorigo, M. (2014). Self-organized task allocation to
634 sequentially interdependent tasks in swarm robotics. *Autonomous Agents and Multi-Agent Systems* 28,
635 101–125. doi:10.1007/s10458-012-9212-y
- 636 Buchanan, E., Pomfret, A., and Timmis, J. (2016). Dynamic Task Partitioning for Foraging Robot Swarms.
637 *International Conference on Swarm Intelligence* 9882, 113–124. doi:10.1007/978-3-319-44427-7_10
- 638 Christensen, A. L., Grady, R. O., and Dorigo, M. (2009). From Fireflies to Fault-Tolerant Swarms of
639 Robots. *IEEE Transactions on Evolutionary Computation* 13, 754–766
- 640 Deb, K., Member, A., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A Fast and Elitist Multiobjective
641 Genetic Algorithm :. *IEEE Transactions on Evolutionary Computation* 6, 182–197
- 642 Delaney, H. D. and Vargha, A. (2000). A Critique and Improvement of the CL Common Language Effect
643 Size Statistics of McGraw and. *Journal of Educational and Behavioral Statistics* 25, 101–132
- 644 Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F., and Martín-Jiménez, M. (2014). Automatic
645 generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition* 47,
646 2280–2292. doi:10.1016/j.patcog.2014.01.005
- 647 Goldberg, D. and Matarie, M. J. (2001). Design and Evaluation of Robust Behavior-Based Controllers for
648 Distributed Multi-Robot Collection Tasks. *Robot teams: from diversity to polymorphism* , 1–24
- 649 Hart, E., Steyven, A. S., and Paechter, B. (2018). Evolution of a functionally diverse swarm via a novel
650 decentralised quality-diversity algorithm. *arXiv preprint arXiv:1804.07655*
- 651 Harwell, J., Lowmanstone, L., and Gini, M. (2020). Demystifying emergent intelligence and its effect on
652 performance in large robot swarms. In *Proceedings of the 19th International Conference on Autonomous*
653 *Agents and MultiAgent Systems*. 474–482
- 654 Hilder, J., Naylor, R., Rizih, A., Franks, D., and Timmis, J. (2014). The Pi Swarm : A Low-Cost Platform
655 for Swarm Robotics Research and Education , 151–162doi:10.1007/978-3-319-10401-0_{_}14
- 656 Humza, R., Scholz, O., Mokhtar, M., Timmis, J., and Tyrrell, A. (2009). Towards energy homeostasis in
657 an autonomous self-reconfigurable modular robotic organism. *Computation World: Future Computing,*
658 *Service Computation, Adaptive, Content, Cognitive, Patterns, ComputationWorld 2009* , 21–26doi:10.
659 1109/ComputationWorld.2009.83
- 660 Keeffe, J. O., Tarapore, D., Millard, A. G., and Timmis, J. (2017). Fault Diagnosis in Robot Swarms : An
661 Adaptive Online Behaviour Characterisation Approach. In *Computational Intelligence (SSCI), 2017*
662 *IEEE Symposium Series* , 1–8
- 663 Li, L., Martinoli, A., and Abu-mostafa, Y. S. (2003). Diversity and Specialization in. *Proc. of the Second*
664 *Int. Workshop on Mathematics and Algorithms of Social Insects.* , 91–98
- 665 Li, X. and Parker, L. E. (2007). Sensor Analysis for Fault Detection in Tightly-Coupled Multi-Robot Team
666 Tasks. In *Proceedings - IEEE International Conference on Robotics and Automation*. April, 3269–3276.
667 doi:10.1109/ROBOT.2009.5152389
- 668 Lu, Q., Hecker, J. P., and Moses, M. E. (2018). Multiple-place swarm foraging with dynamic depots.
669 *Autonomous Robots* 42, 909–926
- 670 McKay, M. D., Beckman, R. J., and Conover, W. J. (1998). A Comparison of Three Methods for Selecting
671 Values of Input Variables in the Analysis of Output From a Computer Code. *Technometrics* 21, 239–245

- 672 Mitchell A. Potter, Lisa A. Meeden, A. C. S. (2001). Heterogeneity in the coevolved behaviors of
673 mobile robots: The emergence of specialists. *International joint conference on artificial intelligence* 17,
674 1337–1343
- 675 Neumann, G., Harman, M., and Poulding, S. (2015). Transformed Vargha-Delaney Effect Size.
676 *International Symposium on Search Based Software Engineering* 1, 318–324. doi:10.1007/
677 978-3-319-22183-0
- 678 Parker, L. E. (1998). ALLIANCE: An architecture for fault tolerant multirobot cooperation. *IEEE*
679 *Transactions on Robotics and Automation* 14, 220–240. doi:10.1109/70.681242
- 680 Parker, L. E. and Kannan, B. (2006). Adaptive causal models for fault diagnosis and recovery in multi-robot
681 teams. In *IEEE International Conference on Intelligent Robots and Systems*. 2703–2710. doi:10.1109/
682 IROS.2006.281993
- 683 Pinciroli, C., Trianni, V., O’Grady, and et al. (2011). ARGoS: A modular, multi-engine simulator for
684 heterogeneous swarm robotics. *IEEE International Conference on Intelligent Robots and Systems* ,
685 5027–5034
- 686 Pini, G., Brutschy, A., Birattari, M., and Dorigo, M. (2011). Task partitioning in swarms of robots:
687 Reducing performance losses due to interference at shared resources. *Lecture Notes in Electrical*
688 *Engineering* 85 LNEE, 217–228. doi:10.1007/978-3-642-19730-7_15
- 689 Pini, G., Brutschy, a., Pinciroli, C., Dorigo, M., and Birattari, M. (2013). Autonomous task partitioning in
690 robot foraging: an approach based on cost estimation. *Adaptive Behavior* 21, 118–136. doi:10.1177/
691 1059712313484771
- 692 Pini, G., Brutschy, A., Scheidler, A., and et al (2014). Task Partitioning in a Robot Swarm: Object Retrieval
693 as Sequence of Subtasks with Direct Object Transfer. *Artificial Life* 20, 291–317. doi:10.1162/ARTL
- 694 Ratnieks, F. L. W. and Anderson, C. (1999). Task partitioning in insect societies. *Insectes Sociaux* 46,
695 95–108
- 696 Tarapore, D., Lima, P. U., Carneiro, J., and Christensen, A. L. (2015). To err is robotic, to tolerate
697 immunological: fault detection in multirobot systems. *Bioinspiration & Biomimetics* 10, 016014.
698 doi:10.1088/1748-3190/10/1/016014
- 699 Timmis, J., Tyrrell, A., Mokhtar, M., Ismail, A., Owens, N., and Bi, R. (2010). An Artificial Immune
700 System for Robot Organisms. *Symbiotic Multi-Robot Organisms: Reliability, Adaptability* , 279–302
- 701 Trianni, V. and Nolfi, S. (2011). Engineering the evolution of self-organizing behaviors in swarm robotics:
702 a case study. *Artificial life* 17, 183–202. doi:10.1162/artl{_}a{_}00031
- 703 Tuci, E. (2014). Evolutionary Swarm Robotics : Genetic Diversity , Task-Allocation and Task-Switching ,
704 98–109
- 705 [Dataset] Winfield, A. F. and Nembrini, J. (2006). Safety in numbers: fault-tolerance in robot swarms.
706 doi:10.1504/IJMIC.2006.008645
- 707 Wold, S., Esbensen, K., and Geladi, P. (1987). Principal component analysis. *Chemometrics and intelligent*
708 *laboratory systems* 2, 37–52
- 709 Zhang, Y., Bastani, F., Yen, I.-L., Fu, J., and Chen, I.-R. (2008). Availability analysis of robotic swarm
710 systems. In *2008 14th IEEE Pacific Rim International Symposium on Dependable Computing (IEEE)*,
711 331–338