# Distributed Task Allocation in Swarms of Robots

**Aleksandar Jevtić**
*Robosoft*
*Technopole d'Izarbel, F-64210 Bidart, France*

**Diego Andina**
*Group for Automation in Signals and Communications*
*E.T.S.I.T.-Universidad Politécnica de Madrid, Ciudad Universitaria, 28040 Madrid, Spain*

**Mo Jamshidi**
*Department of Electrical and Computer Engineering*
*University of Texas, San Antonio, TX 78249 USA*

## ABSTRACT
This chapter introduces a swarm intelligence-inspired approach for target allocation in large teams of autonomous robots. For this purpose, the Distributed Bees Algorithm (DBA) was proposed and developed by the authors. The algorithm allows decentralized decision-making by the robots based on the locally available information, which is an inherent feature of animal swarms in nature. The algorithm's performance was validated on physical robots. Moreover, a swarm simulator was developed to test the scalability of larger swarms in terms of number of robots and number of targets in the robot arena. Finally, improved target allocation in terms of deployment cost efficiency, measured as the average distance traveled by the robots, was achieved through optimization of the DBA's control parameters by means of a genetic algorithm.

## INTRODUCTION
The initial purpose of swarm intelligence algorithms was to solve optimization problems. However, in recent years, these algorithms have shown their full potential in terms of flexibility and autonomy when it comes to design and control of complex systems that consist of a large number of autonomous agents. In more general terms, these can be referred to as systems of autonomous systems [1]. What distinguishes swarm intelligence algorithms in the broad field of soft-computing is that they exploit the decentralizing property of natural swarms in order to create autonomous, scalable, and adaptive multi-agent systems.

Swarm robotics emerged as a straight-forward application domain for swarm intelligence due to resemblance of large robot teams to animal swarms [2-4]. In nature, swarming behavior has been studied in ant and bee colonies, bird flocks and fish schools, among others [5]. However, the biological plausibility of swarm intelligence algorithms, and swarm-based systems in general, is not a must; in computer science and engineering researchers are guided by efficiency, flexibility, robustness and cost as main criteria.

In applications that require area coverage, swarms of mobile robots can use their ability to quickly deploy within a large area. Some of the possible applications include planetary exploration, urban search and

rescue, communication networks, monitoring, surveillance, cleaning, maintenance, and so forth. In order to efficiently perform their tasks, robots require high level of autonomy and cooperation. They use their sensing abilities to explore an unknown environment and deploy on the sites of interest, i.e. targets. However, the coordination of a robot swarm is not an easy problem, especially when the resources for the deployment task are limited. Such a large group of robots, if organized in a centralized manner, could experience information overflow that can lead to the overall system failure [6]. For this reason, the communication between the robots can be realized through local interactions, either directly with one another or indirectly via environment [7].

As a result of the growing interest in the coordination of swarms of robots, multi-robot task allocation (MRTA) has become an important research topic [8, 9]. The goal is to assign tasks to robots in a way that, through cooperation, the global objective is achieved more efficiently. In the scenario proposed in this work, tasks are represented by targets defined with their qualities and their location in the robot arena. For distributed task allocation the Distributed Bees Algorithm (DBA) was proposed [10], which was inspired by the foraging behavior of colonies of bees in nature. In the context of mobile multi-robot systems, scalability refers to the overall system's performance if the number of robots increases in relation to the number of tasks at hand [11]. The resulting effect on the system's performance can be determined in terms of metrics associated with a particular platform or an operating environment, which in this work refers to dispatching a robot to a remote site marked as a target.

The objectives of this chapter are manifold. The following section describes the problem of MRTA and provides a summary of the related work. The Distributed Bees Algorithm (DBA) is then proposed as a solution to distributed MRTA. The DBA's performance is subsequently validated through experiments with physical robots. Moreover, simulator was developed to test the DBA's scalability in terms of number of robots and number of targets. The last experiments present analysis of DBA's performance through optimization of its control parameters. Finally, this chapter provides perspectives on future research directions and gives concluding remarks.

## BACKGROUND

Multirobot systems offer the possibility of enhanced task performance, increased task reliability and decreased cost over more traditional single-robot systems. Various architectures for multirobot systems that differ in size and complexity have been proposed. Dudek et al. [8] provided a taxonomy that categorizes the existing multirobot systems along various axes, including size (number of robots), team organization (e.g., centralized vs. distributed), communication topology (e.g., broadcast vs. unicast), and team composition (e.g., homogeneous vs. heterogeneous).

Rather than characterizing architectures, Gerkey and Matarić [9] categorized instead the underlying coordination problems with a focus on MRTA. The authors distinguish: single-task (ST) and multi-task (MT) robots, single-robot (SR) and multirobot (MR) tasks, and instantaneous (IA) and time-extended (TA) assignment. The authors showed that many MRTA problems can be viewed as instances of well-studied optimization problems in order to analyze the existing approaches, but also to use the same theory in the synthesis of new approaches. In order to estimate a robot's performance, they defined utility that depends on two factors, namely expected quality of task execution and expected resource cost. Given a robot $R$ and a task $T$ one can define $Q_{RT}$ and $C_{RT}$ as the quality and cost, respectively, expected to result from the execution of $T$ by $R$. The resulting nonnegative utility measure is:

*Equation 1. Utility.*

$$U_{RT} = \begin{cases} Q_{RT} - C_{RT}, if\ R\ is\ capable\ of\ executing\ T, and\ Q_{RT} > C_{RT} \\ 0, otherwise \end{cases}$$

This however is not a strict definition of utility which is a flexible measure of performance and can entail arbitrary computation. The only constraint on utility estimators is that they must each produce a single scalar value that can be compared for the purpose of assigning robots for tasks. The problem addressed in this chapter is categorized as a "single-task robots, multirobot tasks, instantaneous assignment (ST-MR-IA)", which Gerkey and Matarić proposed to be solved as a set partitioning problem. However, this requires the combined utilities of all the robots to be known in advance, which is not the case.

What follows is a survey of various multirobot system architectures that have been proposed for solving different problems. We tend to use the above mentioned taxonomies to categorize them.

One of the common approaches for solving the ST-SR and ST-MR problems is a market-based approach which uses auctioning mechanism for task allocation. Matarić et al. [9] proposed four different strategies for dynamical task allocation in two different emergency-handling scenarios. The robots bid for tasks and decisions are made by auctioning. Authors concluded that there is no overall best strategy and that the success of a strategy is task-related. Michael et al. [13] proposed a market-based approach for robots formation control. They associate multiple tasks with predefined spatial locations that define a formation.

A thorough overview of market-based approaches for MRTA is given by Dias et al. [14]. A common drawback of these approaches is the underlying auctioning mechanism which requires all the bids from the robots to be gathered at one auctioning point. Sometimes, when resources permit, markets can even behave in a centralized fashion over larger portions of the robot team to improve solution quality. The authors gave a summary of communication costs for various market-based approaches. The main advantage of the method we propose is that, although it imposes certain communication cost for sending the information of the found targets, the robots make decisions autonomously and in a distributed manner. This is not the case with market-based approaches that feature a partial distribution, where robots are divided into sub-teams that take decisions in a centralized manner. For this reason, scalability in market-based approaches is often limited by the computation and communication needs that arise from increasing auction frequency, bid complexity, and planning demands.

Environment exploration and mapping are common applications for multirobot systems. Franchi et al. [15] proposed a Sensor-based Random Graph (SRG) method for cooperative robot exploration. They addressed the issue of system's performance with respect to exploration time and traveled distance. The authors showed that by adding more robots the system could scale-up, but its performance was highly dependent on the initial team deployment, giving better results when the robots started grouped in a cluster than if scattered in the environment. The robots used broadcast communication with a limited range but the concept of decentralized cooperation was in question since the robots were programmed to gather in sub-teams that had to synchronize for local path-planning and collision-avoidance. This required intensive interchange of information including robot's ID and displacement plan.

Another approach proposed by Burgard et al. [16] treats the unknown environment exploration as a ST-SR problem, where individual robots select a new target location based on its distance and utility. The map was divided in cells whose size was determined by the robot's visual range. The utility of each target location, i.e. cell, would decrease if more of its neighboring cells were assigned to other robots. To determine appropriate target locations for all the robots, the authors proposed an iterative algorithm. The drawback of this algorithm is its complexity and high computational cost. Although the experimental

results show the advantages of collaboration, the proposed centralized approach cannot be applied if not all robots can communicate with each other.

Decentralized coordination of robots has various advantages over more traditional centralized approaches. It can be applied to reduce the communication burden on multirobot system [17], especially for large teams of robots. In some applications communication can be difficult to implement or no communication exists at all. Joordens and Jamshidi [18] proposed a decentralized coordination for a swarm of underwater robots which is based on consensus control. Another decentralized strategy for dynamical allocation of tasks that requires no communication among robots was proposed by Berman et al. [19]. But often, as in case of multirobot area coverage [20], the decentralized coordination and distributed decision-making is applied having one goal in mind, that the global objective is achieved more efficiently.

## Bio-inspired Coordination of Multirobot Systems

Robot swarms are multirobot systems that typically consist of a large population of simple robots interacting locally with one another and with their environment [21]. These systems draw inspiration from animal swarms in nature but their design is not constrained by biological plausibility. Their main feature is decentralized coordination which results in a desired behavior that emerges from the rules of local interactions.

The self-organizing properties of animal swarms such as insects have been studied for better understanding of the underlying concept of decentralized decision-making in nature [22], but it also gave new approach in applications to multi-agent system engineering and robotics. Bio-inspired approaches have been proposed for multirobot division of labor in applications such as exploration and path formation [23], multi-site deployment [24], or cooperative transport and prey retrieval [25, 26].

The bottom-up design topology inherent to bio-inspired multirobot systems provides them with one or more of the following features, such as being autonomous, scalable, robust and adaptive to changes in their environment. On the other hand, the collective behavior has emergent properties that give them the ability to produce unpredictable patterns. One way of dealing with the unpredictability issue is statistical analysis through experiments, as proposed in this chapter.

## Scalability

Task allocation scenarios include a set of tasks that may have different priorities and require one or more robots to be assigned to their execution. A very important property of multirobot systems is the ability to scale-up with respect to the number of robots or the number of tasks at hand. However, scalability of multirobot systems and multi-agent systems in general has been analyzed from various perspectives including the total number of agents involved, the size of the communication data, the number of rules the agents operate with, or the agents' diversity [11].

In order to evaluate the scalability of a given multirobot system one needs to identify a performance metrics. Various MRTA methods exist but, to the best of our knowledge, a comprehensive analysis tool for the scalability of such methods has not been given. Some mathematical models that have been proposed could serve as guidelines in multirobot system design, but different scenarios to which these systems are applied usually do not allow staying within the proposed framework.

Lerman et al. [27] proposed a mathematical model for MRTA in dynamical environments. The authors assumed that robots were able to observe tasks in order to discriminate their types, but also to discriminate the tasks that other robots were assigned to. Robots had limited sensing capabilities and could not directly communicate. The lack of communication made the system more robust to failures, but also more susceptible to noise from the sensors, and requires more time for exploration of available tasks.

Top-down design methodologies apply the classical control theory for performance estimation of distributed agent-based systems. While establishing bounds on the system behavior and provide performance guarantees, they heavily rely on the available bandwidth for robot communication and they are more sensitive to noise. The need for resources becomes even a bigger issue as the number of robots increases. There is therefore a natural tendency to apply bottom-up methodologies that result in autonomous, scalable and adaptable systems requiring minimal communication [28].

Broadcast communication provides quick propagation of tasks' information within the multirobot system but extensive use of communication channel can affect the system's scalability. Previously described market-based approaches suffer from a large requirement in terms of communication bandwidth as they use broadcast messages to auction for the tasks. Farinelli et al. [29] proposed a mechanism based on token passing for cooperative object retrieval, which scales up for reliable sending of broadcast messages. The authors made a comparison of their method with market-based approaches and the ones based on iterative broadcast communication. Their results show that the ability of the system to adjust to the available communication bandwidth provides guarantees for better performance.

## DISTRIBUTED TASK ALLOCATION

### Problem Definition

Based on Dudek's taxonomy, the proposed multi-robot system can be categorized as homogeneous and distributed, using broadcast communication. The problem addressed in this paper is for single-task robots, multi-robot tasks and instantaneous assignment (ST-MR-IA). The task (i.e., target) allocation scenario is placed in a 2-dimensional robot arena with a preset number of targets that could be of same or different importance. A finite number of robots are allowed to be allocated to any target; still, each robot can only be allocated to one target at any given time. Targets have associated quality values and have their own location coordinates. The quality of a target is an application-specific scalar value that may represent target's priority or complexity, where a higher value requires a higher number of allocated robots. (E.g.: In the cleaning of public spaces, this value may represent the amount of detected garbage on site.) The medium by which these values are obtained is not considered in this paper.

The proposed scenario is presented under the following assumptions:

- All the targets are made available to all the robots. This is done by setting a broadcast communication range of the robots to cover the entire arena.
- Robots take decision once a predefined number of targets in the arena are found. The robots that found a target are automatically allocated to that target.
- Reallocation to another target is not allowed.

These assumptions are taken for simplicity; otherwise, it would be difficult to analyze the performance of the system due to the unpredictability of the robots' distribution prior to target allocation. It is important to mention that the entire swarm is involved in the search for targets. The search phase was used in experiments with physical robots in order to estimate the odometry error rate with respect to the experiment execution time. It was later used in simulation in order to have a realistic scenario. The experimental setup has a limitation that the robots wait for a preset number of targets to be found in order to allocate. This value can be altered or set as a variable, but that is not considered in this study and remains to be a part of future work. Even though the broadcast communication represents a centralized solution, the decision making is executed by the robots in a distributed manner, which is an inherent characteristic of swarms in nature.

The MRTA problem can be described as follows. Consider a population of $N$ robots to be allocated among $M$ targets ($N \geq M$). Let $Q \in \{q_1,\dots,q_M\}$ denote the set of normalized qualities of all available targets. We denote the number of robots on the target $i \in \{1,\dots,M\}$ by $n_i$, a nonnegative integer. The population fraction allocated to target $i$ is $f_i = n_i/N$, which represents the target's relative frequency, and the vector of population fraction is $\mathbf{f} = [f_1,\dots,f_M]^T$. The expected distribution is the set of desired population fractions for each task, $\mathbf{f^d} = [f_1^d,\dots,f_M^d]^T$, where $f_i^d = q_i$. The usage of fractions rather than integers is practical for scaling, but it also introduces a distribution error as the fractions can take only certain values that are defined by the swarm size.

A relevant concept from set theory could be used to observe this as a set partitioning problem. A family $X$ is a partition of a set $E$ if and only if the elements of $X$ are mutually disjoint and their union is $E$:

*Equation 2. Partition Set.*

$$\bigcap_{x \in X} = \emptyset$$

$$\bigcup_{x \in X} = E$$

However, for the proposed scenario the system optimization based on the maximum utility cannot be applied because the combined utilities of the robots are unknown as robots have no knowledge of the decisions taken by other robots. Therefore, the DBA is proposed.

## Distributed Bees Algorithm

The DBA [30] was applied to multi-robot target allocation in the proposed scenario. The robots start a search for the targets from their randomly chosen initial positions in the arena. When a robot finds a target, it broadcasts the message containing the target's quality. When another robot receives information on the predefined number of targets, it calculates the utilities with respect to those targets. The utility depends on the target's quality value and the related deployment cost measured as the robot's distance from the target. The distance to the target is obtained thanks to a local, distributed and situated communication system [31, 32]. When a robot broadcasts information about the target, a receiver robot obtains the information transmitted together with the range (distance) and bearing (orientation) to the emitter robot. Therefore, the robot is able to calculate the distance and orientate to the emitting robot. The main concepts behind the implementation of the DBA are presented hereafter.

The cost of a target $i$ for robot $k$ is calculated as the Euclidean distance, $d_i^k$, between the robot and the target in a two-dimensional arena. However, the target's visibility is defined as the reciprocal value of the distance:

*Equation 3. Visibility.*

$$\eta_i^k = \frac{1}{d_i^k}$$

The target's quality is a scalar value that represents its priority or complexity. Normalized qualities are calculated as fractions of the sum of qualities:

*Equation 4. Normalized target qualities.*

$$q_i = \frac{Q_i}{\sum_{j=1}^{M} Q_j}$$

where $Q_i$ is a quality of the target $i$. In real-world scenarios, the quality of a region of interest is an estimated value that is as a result of sensor-readings or a previously acquired knowledge.

The utility of a robot depends on both visibility and quality of the chosen target. The utility is defined as a probability that the robot $k$ is allocated to the target $i$, and it is calculated as follows:

*Equation 5. Allocation probabilities.*

$$p_i^k = \frac{q_i^\alpha \eta_i^\beta}{\sum_{j=1}^{M} q_j^\alpha \eta_j^\beta}$$

where $\alpha$ and $\beta$ are control parameters that allow biasing of the decision-making mechanism towards the quality of the solution or its cost, respectively. ($\alpha, \beta > 0$; $\alpha, \beta \in \mathbf{R}$.) From (5) it is easy to show that

*Equation 6. Sum of the probabilities.*

$$\sum_{j=1}^{M} p_i^k = 1$$

The underlying decision-making mechanism of the DBA algorithm adopts the roulette rule, also known as the wheel-selection rule. That is, every target has an associated probability with which it is chosen from a set of available targets. Once all the probabilities are calculated as in (5), the robot will choose a target by "spinning the wheel".

It should be noticed that the resulting robots' distribution depends on their initial distribution in the arena, i.e. their distances from each target prior to target allocation. Therefore, robots' utilities will differ with respect to the same target if their distances from that target are not equal. Since a combined robots utility cannot be computed due to a distributed nature of the proposed algorithm, the quality of the targets is used as the only measure for the expected robots' distribution. Although the overall cost efficiency of the swarm is not analyzed here, target's visibility as used in (5) makes closer targets more attractive to robots.

## EXPERIMENTAL EVALUATION

In this section, the results from the experiments with physical robots are presented to validate the performance of the proposed DBA. The overview of robots' hardware and the communication protocol used in experiments is given, followed by the description of the experimental setup. The experiments results are presented and discussed.

### Robot hardware

The robots were assembled with the same selection of hardware components (see fig. 1). The Lynxmotion Terminator Sumo Robot Kit with four Spur Gear Head Motors was used as a base to build the robots, although any platform that could support the listed hardware components would be suitable. The DC

motors were powered with 12 VDC, with 200 RPM, torque of 63.89 oz.in (4.6 Kg-cm), 30:1 reduction and 6 mm shaft diameter, and they were paired as left-hand and right-hand, in order to be able to perform rotation on-the-spot. Devantech MD22 Motor Driver was used to control the motors' rotation speed and direction. It averaged the PWM signal received from the Arduino microcontroller board to provide a proportional value of the 12 VDC from the battery. The four switches on the motor driver's board were used to define the working mode. In the experiments, the analog mode was used which provided satisfactory speed control.

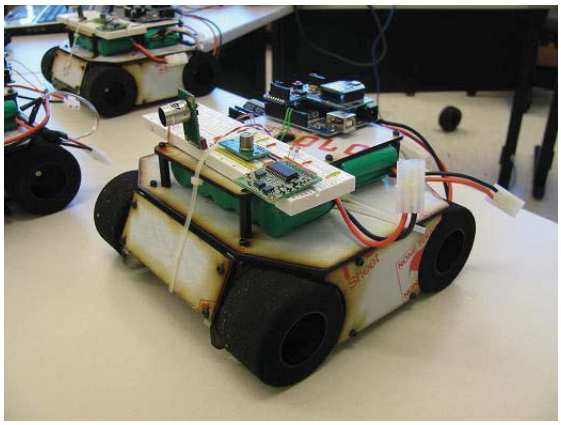


*Figure 1. Sumo robot used in the experiments.*

Arduino Duemilanove microcontroller board with ATMEGA328 microcontroller was powered with the 6 VDC battery. ZigBee module used for robot communication was connected using the Arduino Xbee shield. Sensors were programmed for $I^2C$ communication protocol with the microcontroller. One ultrasonic sensor was mounted on each robot for obstacle detection. Thermal sensor was used to detect the targets. The odometry error inherent to all mobile robots affected their precise localization. It cannot be eliminated but various methods for its reduction have been proposed, such as the averaging method proposed by Gutierrez et al. [33].

## Coordinator and communication

A computer with a Pentium IV processor at 3 GHz with 2 GB of RAM was used to program the robots and to connect a ZigBee communication module that created a mesh communication network (the coordinator). The ZigBee modules mounted on robots were able to detect a reserved communication channel and connect with the coordinator. This allowed the communication between the robots and the robots with the computer. The broadcast mode that allowed each module to communicate with any other module in the network was used.

## Experimental Results

The main objective of the experimental setup was to test the performance of the proposed algorithm and not the sensing and pattern recognition capabilities of the robots. The robots estimated their position based on the speed, time and direction of displacement. Although the robots performed well in detecting the heat source and sending the estimated location and measured temperature, which was tested in initial experiments, in order to test the performance of the DBA algorithm a simplified scenario was arranged. A small swarm of three real and two simulated robots was used in search of two targets.

The experiments were performed in 12x12 sq ft (3.65x3.65 $m^2$) arena, with randomly distributed obstacles. Robots were placed at the preprogrammed initial locations. When the command was sent from the coordinator the robots started the random search. After a certain period of time $t_0$, the information of

two targets was sequentially sent from the coordinator. The information included the targets' estimated locations and their temperature (quality) values. The robots calculated the probabilities to move to each target as in (5). The physical robots are unaware of the message sender identity, so the coordinator was able to simulate two robots that found two different targets.

Two types of experiments were performed. In the first one, the random search time $t_0$ was changed to test the increase rate of the odometry error over time. Single robot was used in in each run in order to avoid collision with other robots, and in $t_0$ the information about the found target was sent from the coordinator. Obstacles were introduced the arena to force the robot to change its motion. Simple obstacle detection algorithm was implemented where the robot randomly rotates left or right by 90 degrees and then continues moving forward. With each of three robots 30 experiments were conducted in order to obtain the average odometry error value. The results of the first experimental setup are shown in fig. 2. The search was considered successful if the robot was able to get as close as 30.48 cm (1 ft) from the target. It can be noted that while increasing the initial random search time of the robot the odometry error increased as well. This happened due to the imperfect calibration of the DC motors, non-constant battery voltage, friction of the ground, etc. It can also be noticed that for the $t_0$ < 90s the experiment success rate was 100 %.

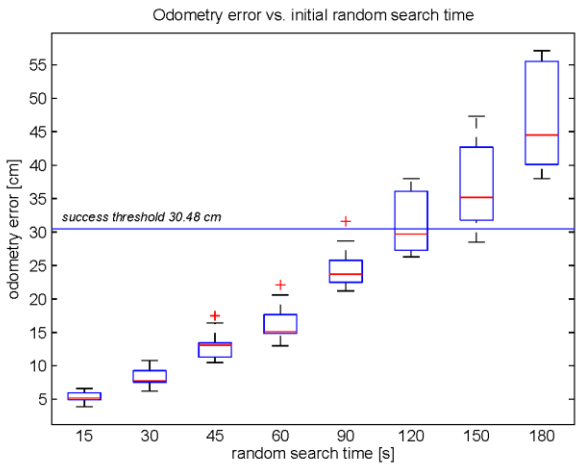



*Figure 2. Results of the first experimental setup: Average odometry error vs. initial random search time.*

The results from the first experimental setup were used to set the parameters for the second experimental setup. The random search initial time value was set to $t_0$ = 30s, which guaranteed that the odometry error would be below the success threshold. The scenario involved all three robots in search for two targets whose information was sent form the coordinator. The choice of having two targets in the scenario was based on the number of real robots that were at our disposal. By having three robots and two targets we could test how the robots distributed in the arena based on the targets' quality values. Four possible events could occur: 1) all robots go for the first target (T1); 2) all robots go for the second target (T2); 3) two robots go for the target T1 and one for the target T2; and 4) one robot goes for the target T1 and two robots go for the target T2.

Because of the relatively small size of the robot arena, the visibility of the targets was set to be constant, $\eta_1 = \eta_2 = 1$, and the target allocation was performed based only on the targets quality values. The values of the control parameters were also set to $\alpha = \beta = 1$. In order to test the self-organized behavior of the swarm of robots, the quality values of the two targets sent from the coordinator to the robots were changed. The experimental results are shown in Table 1. It can be noticed that when the quality values were equal, $Q_1 = Q_2 = 50$, in most cases two robots would go for one target and one would go for the

other. This was expected since the probabilities of choosing any of the targets were equal. By increasing the difference between the quality values, the distribution would change in favor of the target with the higher quality value because the probability that a robot chooses that target also increased.

*Table 1. Robots distribution vs. targets quality values.*

| Quality values | T1:T2 | Occurence | Occurrence [%] |
|---|---|---|---|
| $Q_1=Q_2=50$ | 2:1 | 11 | 36.67 |
| $Q_1=Q_2=50$ | 3:0 | 4 | 13.33 |
| $Q_1=Q_2=50$ | 1:2 | 13 | 43.33 |
| $Q_1=Q_2=50$ | 0:3 | 2 | 6.67 |
| $Q_1=70;Q_2=30$ | 2:1 | 18 | 60.00 |
| $Q_1=70;Q_2=30$ | 3:0 | 9 | 30.00 |
| $Q_1=70;Q_2=30$ | 1:2 | 2 | 6.67 |
| $Q_1=70;Q_2=30$ | 0:3 | 1 | 3.33 |
| $Q_1=90;Q_2=10$ | 2:1 | 8 | 26.67 |
| $Q_1=90;Q_2=10$ | 3:0 | 21 | 70.00 |
| $Q_1=90;Q_2=10$ | 1:2 | 1 | 3.33 |
| $Q_1=90;Q_2=10$ | 0:3 | 0 | 0.00 |

The experimental results show that the task allocation was performed according to the targets' quality values in an autonomous and decentralized manner. The targets with higher quality values attracted more robots, which was the objective for the multi-foraging scenario. The odometry error inherent to mobile robots was used as an advantage in order to gather the robots in the vicinity of the found targets and not at their exact locations. Still, there is a necessity to maintain the odometry error within the acceptable limits, and this is planned as a part of the future work.

## EVALUATION THROUGH SIMULATIONS

The experiments with real robots could not provide the insight on the multirobot system's scalability because of the small number of available robots. Therefore, the experiments were performed in a simulated environment which provided the results for a thorough analysis of the algorithm's performance. In this section, the simulator and the simulation setup are described, and the simulation results are presented in order to analyze the scalability of the DBA.

### Simulator

Simulation platform is a fast, specialized multi-robot simulator for the e-puck robots described in [33]. It is a simple and effective simulator implementing 2D kinematics. A screenshot of the simulator is shown in fig. 3. Simulator screenshot shows 40 robots engaged in search for 4 targets of different qualities represented by different grey-level intensity. Robots are programmed for obstacle avoidance; when robot detects an obstacle its color changes from black to blue to mark his new state. Once the robot has taken a new direction, its color goes back to black. In simulations, the e-puck is modeled as a cylindrical body of 3.5 cm in radius that holds 8 infrared (IR) proximity sensors distributed around the body, 3 ground sensors on the lower-front part of the body and a range and bearing communication sensor. IR proximity sensors have a range of 5 cm, while the communication range of the E-puck Range&Bearing module was set to cover the whole arena. For the three types of sensors, real robot measurements were sampled and the data was mapped into the simulator. Furthermore, uniformly distributed noise was added to the samples in order to effectively simulate different sensors; +/- 20% noise is added to the IR sensors and +/- 30% to the ground sensors. In the range and bearing sensor, noise is added to the range (+/- 2.5 cm) and bearing (+/- 20°) values. A differential drive system made up of two wheels is fixed to the body of the

simulated robot. At each time step of 100 ms, the robot senses the environment and actuates. The robot's speed was limited to 6 cm/s when moving straight and 3 cm/s when turning.

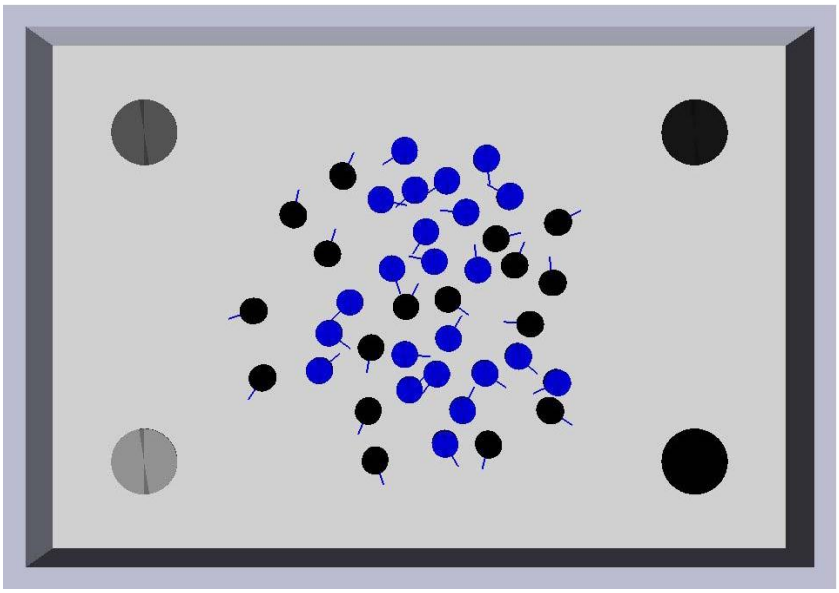


*Figure 3. Simulator screenshot.*

## Simulation Setup

Three different simulation setups have been chosen to compare and study performance and scalability of the proposed DBA algorithm. The setups were carried out in the same arena where the number of robots, number of targets and targets' quality values were changed as shown in Table 2. Additional simulation setup was created in order to analyze the effect of the control parameters $\alpha$ and $\beta$ on the resulting distribution. Each simulation was repeated 50 times for different initial robot distribution in order to perform an analysis of the results.

*Table 2. Parameters describing three arenas used in simulations.*

| | Arena 1 | | | | | Arena 2 | | | | | Arena 3 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Area dimensions [$m^2$] | 1.5x2.125 | | | | | 1.5x2.125 | | | | | 1.5x2.125 | | | | |
| Number of robots | 10 | 20 | 40 | 60 | 100 | 10 | 20 | 40 | 60 | 100 | 10 | 20 | 40 | 60 | 100 |
| Simulation duration [time steps] | 400 | 400 | 400 | 300 | 200 | 400 | 400 | 400 | 300 | 200 | 400 | 400 | 400 | 300 | 200 |
| Time step duration [s] | 0.1 | | | | | 0.1 | | | | | 0.1 | | | | |
| Initial area radius [m] | 0.4 | 0.4 | 0.4 | 0.4 | 0.5 | 0.4 | 0.4 | 0.4 | 0.4 | 0.5 | 0.4 | 0.4 | 0.4 | 0.4 | 0.5 |
| Number of targets | 2 | | | | | 4 | | | | | 4 | | | | |
| Target radius [m] | 0.09 | | | | | 0.09 | | | | | 0.09 | | | | |
| Target 1 location ($x$,$y$) [m] | (-0.45,0.75) | | | | | (-0.45,0.75) | | | | | (-0.45,0.75) | | | | |
| Target 2 location ($x$,$y$) [m] | (0.45,-0.75) | | | | | (0.45,-0.75) | | | | | (0.45,-0.75) | | | | |
| Target 3 location ($x$,$y$) [m] | N/A | | | | | (-0.45,-0.75) | | | | | (-0.45,-0.75) | | | | |
| Target 4 location ($x$,$y$) [m] | N/A | | | | | (0.45,0.75) | | | | | (0.45,0.75) | | | | |
| Target 1 quality ($q_1$) | 0.5 | | | | | 0.25 | | | | | 0.1 | | | | |
| Target 2 quality ($q_2$) | 0.5 | | | | | 0.25 | | | | | 0.2 | | | | |
| Target 3 quality ($q_3$) | N/A | | | | | 0.25 | | | | | 0.3 | | | | |
| Target 4 quality ($q_4$) | N/A | | | | | 0.25 | | | | | 0.4 | | | | |

## Simulation Results and Discussion

In order to test the scalability of the proposed DBA with respect to the size of the swarm, the simulations were performed with 10, 20, 40, 60 and 100 robots for the simulation setup 1, and 20, 40, 60 and 100 robots for the simulation setup 2 and the simulation setup 3. The number of targets was also changed, from two in the simulation setup 1 to four in the simulation setup 2, in order to test the performance of the algorithm with respect to the number of targets. In the simulation setup 3, four targets with different quality values were used in order to show the adaptability of the swarm to a non-uniform distribution of the "food" in the environment. This is also the most realistic scenario. Finally, the simulation setup 4 was created to test how the change in the ratio of the control parameters $\alpha$ and $\beta$ can affect the resulting robots' distribution.

As the algorithm performance metrics the mean absolute error (*MAE*) of the robots' distribution was used, given by

*Equation 7. MAE of the robots' distribution.*

$$MAE = \frac{1}{M}\sum_{i=1}^{M}\left|f_i - f_i^d\right|$$

where $f_i^d = q_i$.

As the name suggests, the *MAE* is the average value of the absolute distribution error (per target) that is the result of discrepancy between the expected and the resulting robots' distribution. For each simulation setup and each swarm size described in Table 2. fifty simulations were performed. The values of *MAE* obtained from the simulations are graphically shown in fig. 4. It can be noticed that the average *MAE* and maximum *MAE* values decrease as the size of the robot swarm increases regardless of the number of targets or their quality values. This was expected because of the probabilistic target allocation mechanism applied in (5). The results from the simulation setup 1, 2, and 3, are shown in fig. 5, 6, and 7, respectively.

The effectiveness of the algorithm in terms of increased number of targets can be seen from the results shown in fig. 5 and 6. The results show that the average and the maximum *MAE* values decreased for larger swarms in case of 4 targets of the same quality. It should be noticed that the allocation of 10 robots to 4 targets produces an error that is the result of the cardinality of the robot swarm. It is not physically possible to partition the swarm in order to obtain the expected target allocation (2.5 robots per target).

Another inherent source of error results from the assumption that the robots that had found a target are not allowed to reallocate to another target, therefore they are not involved in the decision-making process. Also, it is assumed that the robots wait for a predetermined number of targets to be found before they make a decision, which can result in the same target being found by more than one robot. This fraction of the robot swarm also produces an error in the final distribution because they cannot reallocate to another target. The algorithm's performance is analyzed having these issues mind.

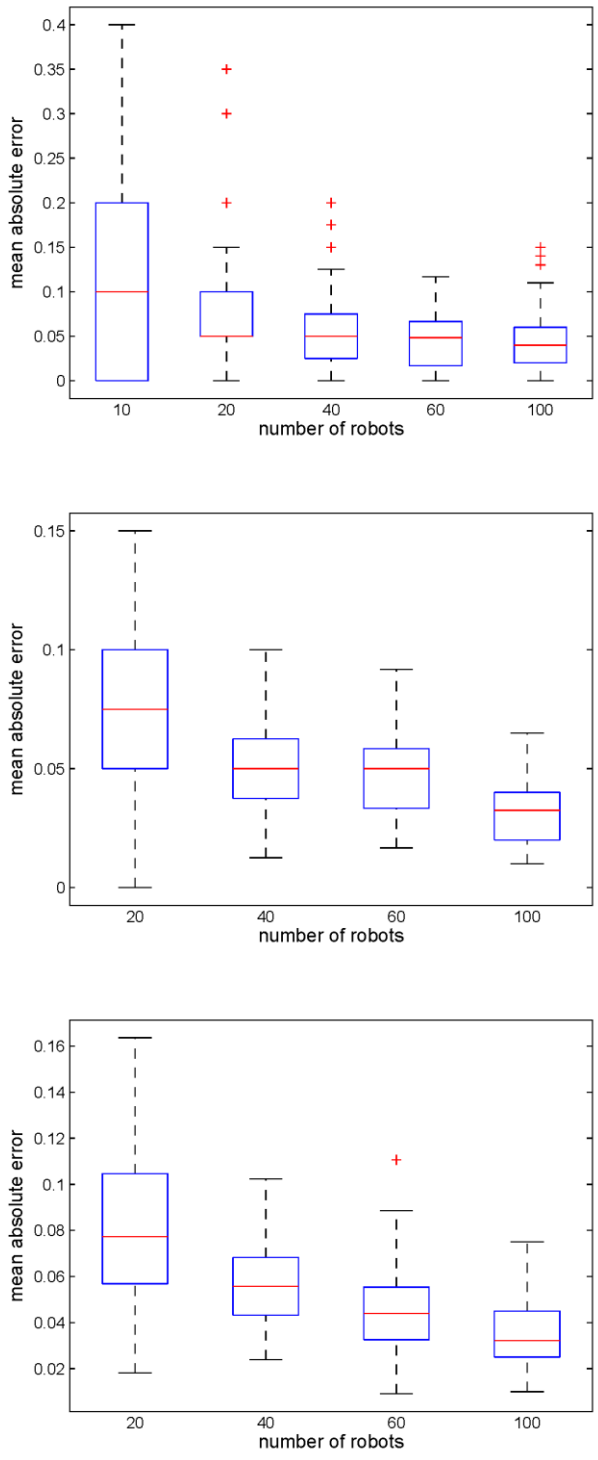



*Figure 4. Distribution MAE with respect to the swarm size: a) simulation setup 1; b) simulation setup 2; and c) simulation setup 3. Each box-plot comprises observations ranging from the first to the third quartile. The median is indicated by a horizontal bar, dividing the box into the upper and lower part. The whiskers extend to the farthest data points that are within 1.5 times the interquartile range. Outliers are shown with a plus symbol. The values were obtained from 50 simulations performed for each swarm size within each simulation setup.*

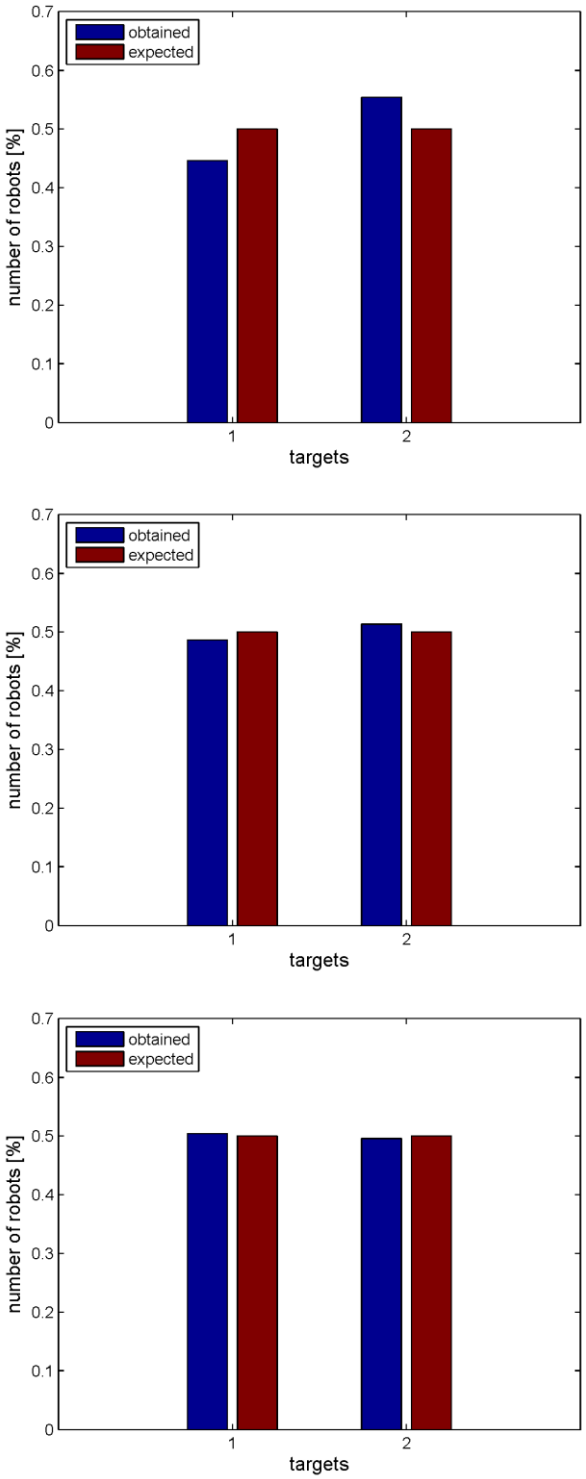



*Figure 5. Expected vs. obtained robots distribution for two equal targets. Targets quality values are $q_1 = q_2 = 0.5$. Fifty simulations were performed for each of the following swarm sizes: a) 10 robots; b) 40 robots; and d) 100 robots.*

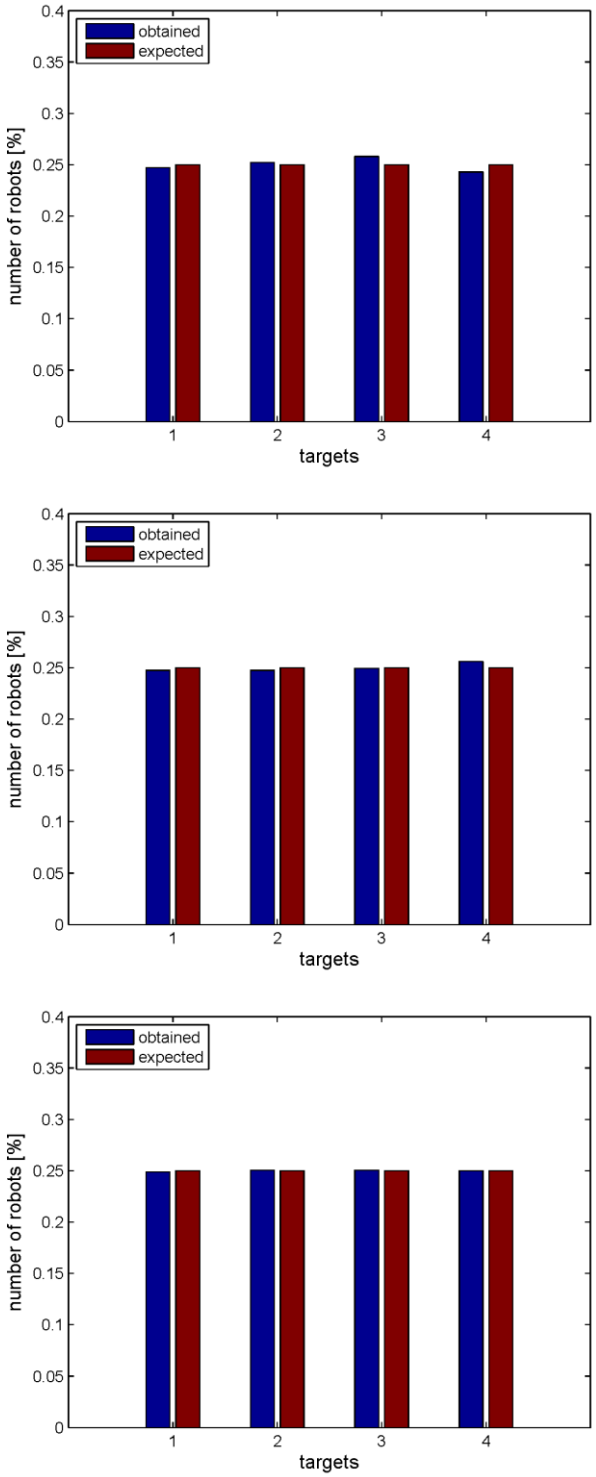



*Figure 6. Expected vs. obtained robots distribution for four equal targets. Target quality values are $q_1 = q_2 = q_3 = q_4 = 0.25$. Fifty simulations were performed for each of the following swarm sizes: a) 20 robots; b) 60 robots; and d) 100 robots.*

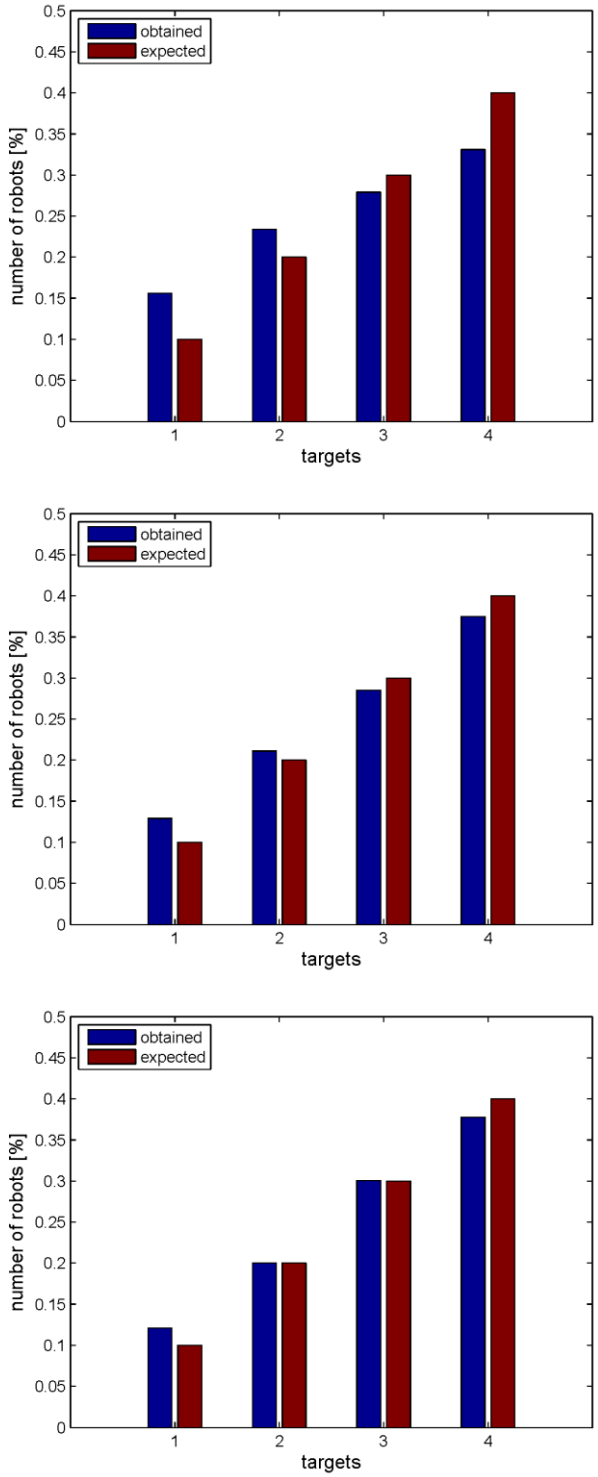



*Figure 7. Expected vs. obtained robots distribution for four different targets. Target quality values are $q_1 = 0.1$, $q_2 = 0.2$, $q_3 = 0.3$, and $q_4 = 0.4$. Fifty simulations were performed for each of the following swarm sizes: a) 20 robots; b) 60 robots; and d) 100 robots.*

In order to test the ability of the robot swarm to adapt to a non-uniform distribution of "food" in the environment, the simulations were performed for four different targets (simulation setup 3). The robots' distribution changed according to a new set of targets' quality values, as shown in fig. 7. It can be noticed that the resulting distribution is slightly in favor of the less valuable targets. This is another consequence of the robots that had found a target not being able to reallocate, and it is especially evident for smaller robot swarms. For example, let's consider a swarm of 10 robots in search of 4 different targets, as shown in fig. 7. If in the random target search process two robots find the target with the associated quality value of 0.1, then the final relative frequency for this target cannot be less than 0.2 (2 out of 10 robots) which is already above the expected value of 0.1. Although for the larger swarms the effect of the initial robot distribution becomes less relevant, it is always present.

The control parameters $\alpha$ and $\beta$ were introduced in (5) to compensate for the biased distribution, but also to give more relevance to either the quality of the targets or to the cost of reaching them. In the simulation setup 4, the $\alpha/\beta$ ratio was increased to give more relevance to the quality value of the targets on the expense of their distances from the robots. The resulting robots' distributions per target for different values of the $\alpha/\beta$ ratio are shown in fig. 8. Results show that, by tuning the control parameters, the final robot distribution can change in favor of the more valuable targets but with an increase in the average *MAE* (see Table 3.). It is reasonable to expect that by decreasing the $\alpha/\beta$ ratio the cost efficiency of the robot swarm would improve in terms of the distance traveled, however, the *MAE* is also expected to increase. More detailed analysis of the effect of the control parameters is given in the following section.

*Table 3. Effects of control parameters on robots' distribution.*

| $\alpha/\beta$ ratio | Average *MAE* | Maximal *MAE* |
|---|---|---|
| 1 | 0.0478 | 0.1083 |
| 2 | 0.0525 | 0.1000 |
| 5 | 0.1415 | 0.2083 |

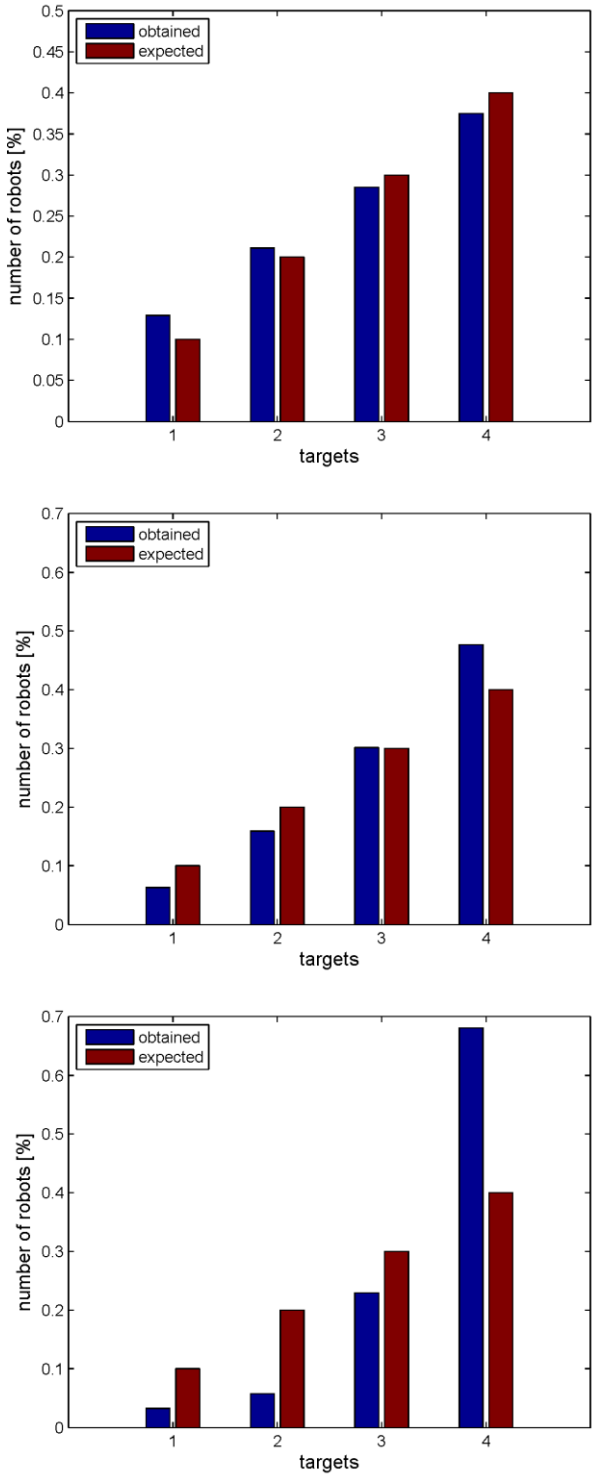



*Figure 8. Effects of the DBA's control parameters on the final robots' distribution. Target allocation was performed with 60 robots as described in the simulation setup 3 consisting of 4 targets with different quality values: $q_1 = 0.1$, $q_2 = 0.2$, $q_3 = 0.3$, and $q_4 = 0.4$. The results of the robots' distribution per target are shown for the following values of $\alpha/\beta$ ratio: a) $\alpha/\beta=1$; b) $\alpha/\beta=2$; c) $\alpha/\beta=5$. The values were obtained from 50 simulations for each scenario.*

## TUNNING OF CONTROL PARAMETERS

The probability function in (5) introduced a set of control parameters $\alpha$ and $\beta$ that can be used to adapt swarm's behavior for different operational objectives. In this work, the parameters are optimized to improve target allocation in terms of deployment cost. Deployment cost is measured as the average distance traveled by the robots in the deployment stage. By changing these parameters' values, robots distribution patterns can be modified. The parameters were optimized by means of a genetic algorithm (GA) [34]. GAs have proven to be powerful optimization tools. They are population-based algorithms, which means that they create a population of solutions (genes) in the data space in order to avoid getting stuck in a local optimum [35].

### Genetic Algorithm

In the DBA optimization both control parameters have been taken into account. The parameters $\alpha$ and $\beta$ define how distances (i.e. visibilities) and quality values affect distribution of the robots in the arena. The effect of the parameters is exponential; hence, a small change in their values can result in very different robots' distribution patterns, and a larger distribution error. Moreover, since a large number of agents can be found in a swarm, increase in the deployment cost can be significant. Therefore, even though a simple sampling of the solution space would be less computationally demanding, in order to obtain a high accuracy and considering that the parameters optimization is performed offline, a GA was used.

In order to limit the complexity of the exploration process, the following range of possible values was defined for both parameters: $\alpha, \beta \in (0, 5]$. Initially, a population of 30 random genotypes was created, in which values are drawn from uniform distributions in the respective ranges of the parameters. The genetic algorithm was run for 1000 generations, during which new generations of genotypes were bred. The genetic algorithm loop consists of the evaluation, the selection and the reproduction of the genotypes. In order to evaluate the fitness of a given genotype, the controller of 40 simulated robots was parameterized with the values of $\alpha$ and $\beta$ encoded in the genotype. The total number of 50 simulated experiments was run with different initial conditions. The experiments duration was set to 100 s.

The fitness function *F(g)*, of the evaluated genotype *g*, is computed as an indicator of the swarm's ability to allocate the robots according to the targets' qualities ($q_i$) and visibilities ($\eta_i$). The fitness *F* is defined as follows:

*Equation 8. Fitness function.*

$$F = \frac{1}{MAE \cdot d_a}$$

where *MAE* is the mean absolute distribution error and $d_a$ is the average distance traversed by all the robots.

Generations following the first one are produced by a combination of selection with elitism, recombination and mutation. For each new generation, the two highest scoring individuals ("the elite") from the previous generation are retained unchanged. The remainder of the new population is generated by fitness-proportional selection (also known as roulette selection) from the individuals of the old population. Mutation entails that a random Gaussian offset is applied to each real-valued vector component encoded in the genotype (except the elite), with a probability of 0.5. The mean of the Gaussian is $\mu = 0$, and its standard deviation is $\sigma = 0.1$. During evolution, all vector component values are constrained to remain within the range [0, 1]. Once the new population has been created, the genotype parameters are linearly mapped to produce network parameters within the aforementioned ranges.

## Simulation setup

The simulation setup 1, 2, and 3 compare the swarm's performance for the new and the initially-used set of $\alpha$ and $\beta$ values [30], $\alpha = \beta = 1$. These sets are referred to as optimal and non-optimal, in terms of deployment cost. The range of parameters' values is shown in the Table 4.

In the simulation setup 1, the system's robustness was tested with respect to the change of the swarm's size. The number of robots was varied, and the targets' position and quality values were preset. In the simulation setup 2, the size of the robot arena was varied to avoid specialization of the system for a specific environment. Finally, in the simulation setup 3, the performance of the system was tested with respect to different distribution of the targets of random quality values. These simulation setups were proposed in order to perform an indebt system's performance analysis.

*Table 4. Parameters describing simulation setups 1, 2, and 3.*

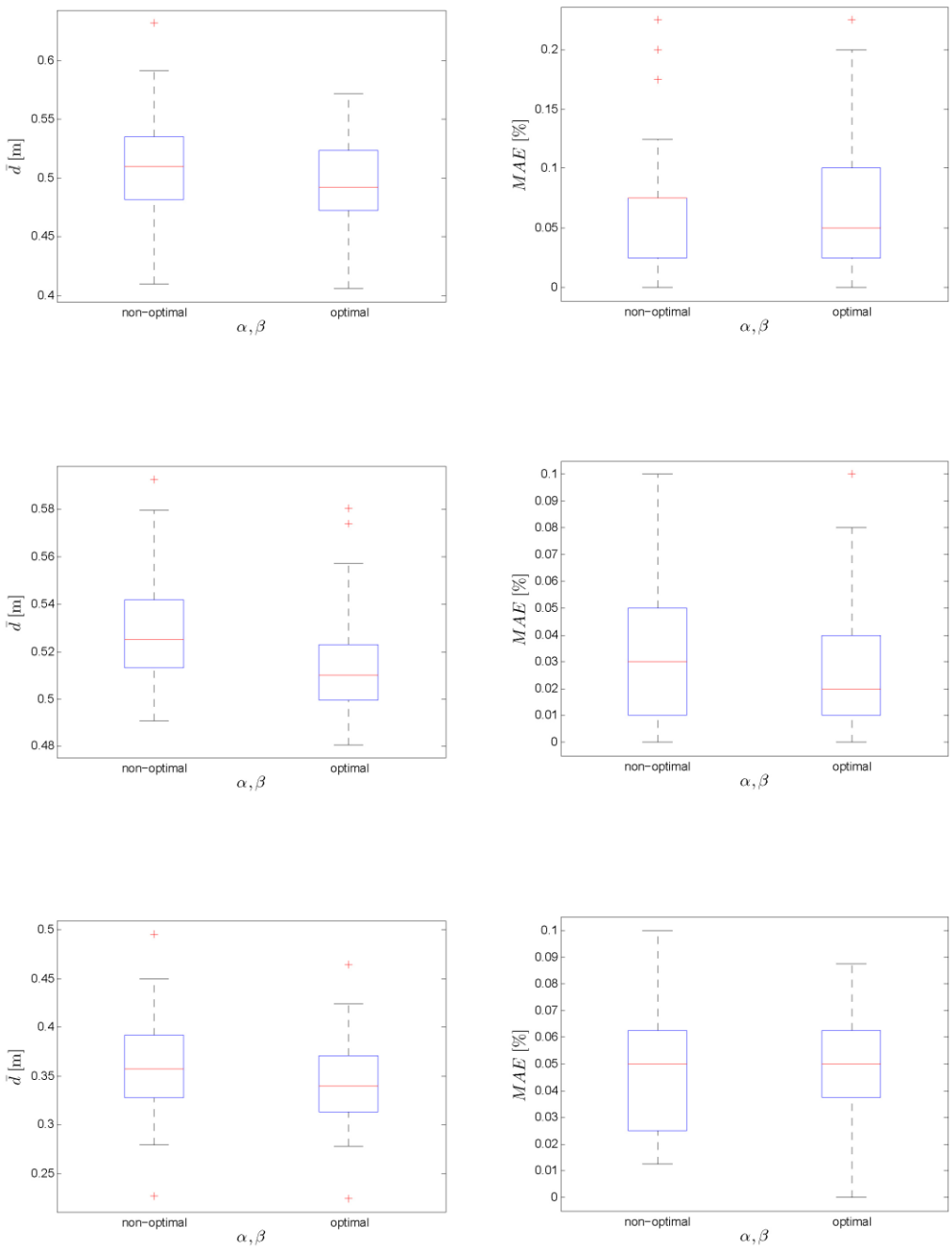
| Parameter | Values range |
|---|---|
| $\alpha$ | 1, 2.65 |
| $\beta$ | 1, 2.55 |
| Area dimension [$m^2$] | 2.25x3.1875, 3.0x4.25, 4.5x6.375, 6.0x8.5 |
| Number of robots | 40, 100 |
| Simulation duration [time steps] | 100, 200, 300 |
| Time step duration [s] | 0.1 |
| Initial area radius [m] | 0.4, 0.5 |
| Number of targets | 2, 4 |
| Target radius [m] | 0.09 |
| Target location ($x$,$y$) [m] | fixed, random |
| Target qualities ($q$) | fixed, random |

## Simulation results and discussion

In this subsection, the results from three proposed simulation setups are presented and discussed. Each simulation was repeated 50 times.

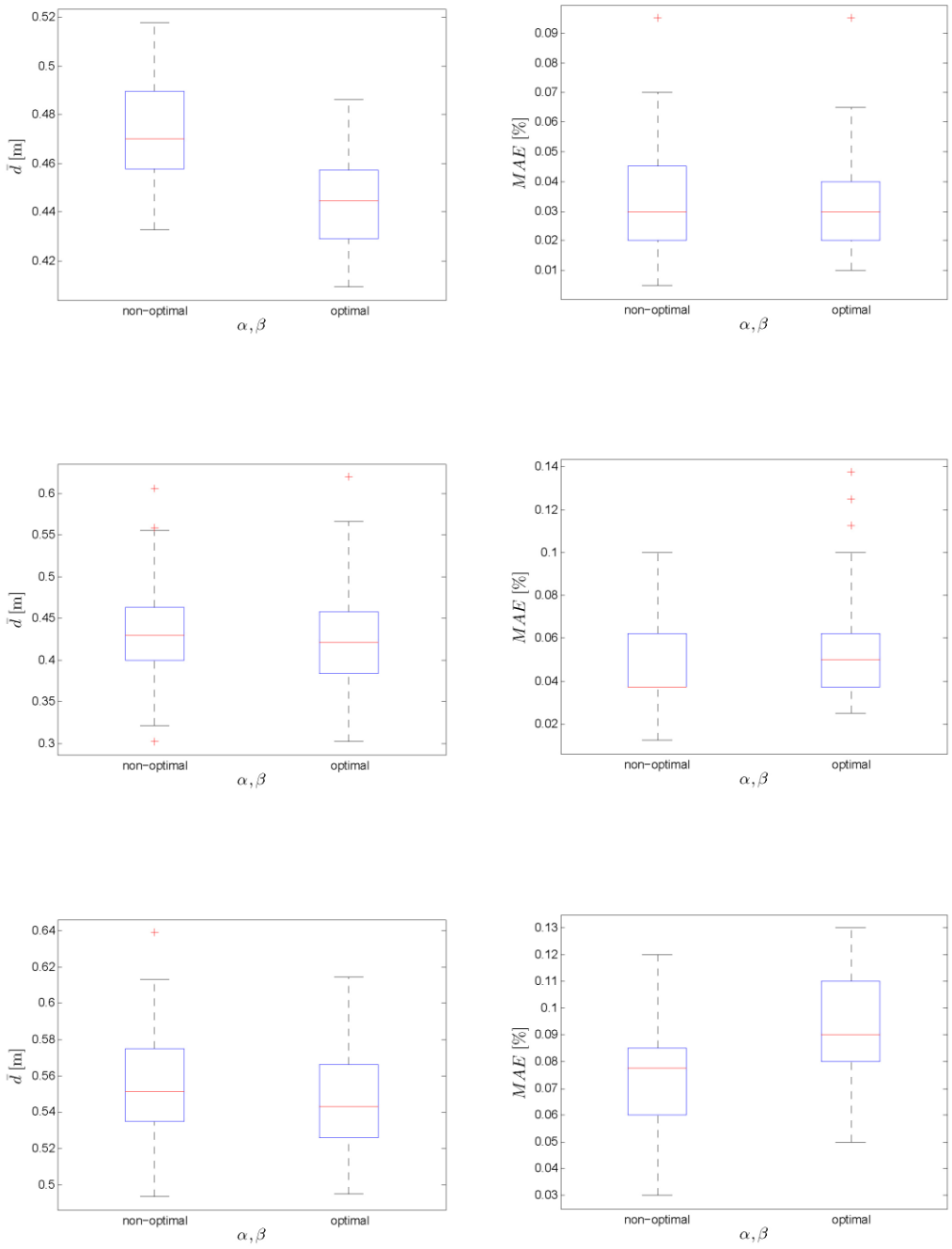
The simulation setup 1 was proposed to test the swarm's performance when the number of robots (40 and 100) and the number of targets (2 and 4) were changed. The targets' associated quality values were set to $q_1 = q_2 = 0.5$ and $q_1 = q_2 = q_3 = q_4 = 0.25$ for 2 and 4 targets, respectively. Additional experiment was performed with 4 targets that had different, but predefined, associated qualities $q_1 = 0.1$, $q_2 = 0.2$, $q_3 = 0.3$, $q_4 = 0.4$. In order to measure the swarm's performance, median distance value and mean absolute robot distribution error were used. The experimental results for non-optimal ($\alpha = \beta = 1$) and optimal ($\alpha = 2.65$, $\beta = 2.55$) set of values are shown for 40-robot and 100-robot swarms in fig. 9 and fig. 10, respectively. It can be noticed that with the optimal set of control parameters swarm obtains more efficient distribution at a lower deployment cost. Only in case of 100 robots in a search of 4 targets, with equal or different qualities, the deployment cost was decreased at the expense of a higher distribution error.

This simulation setup 2 tests swarm's performance in case of a random distribution of the targets in the arena. Four arenas that differ in size were used (see Table. 4). The scenario involved 100 robots in the search for 4 different targets with predefined quality values $q_1 = q_2 = q_3 = q_4 = 0.4$. It can be noticed from the fig. 11 that in all the simulations the optimal control parameter values improved the performance of the swarm with respect to the deployment cost. This was achieved at the expense of a higher distribution error.



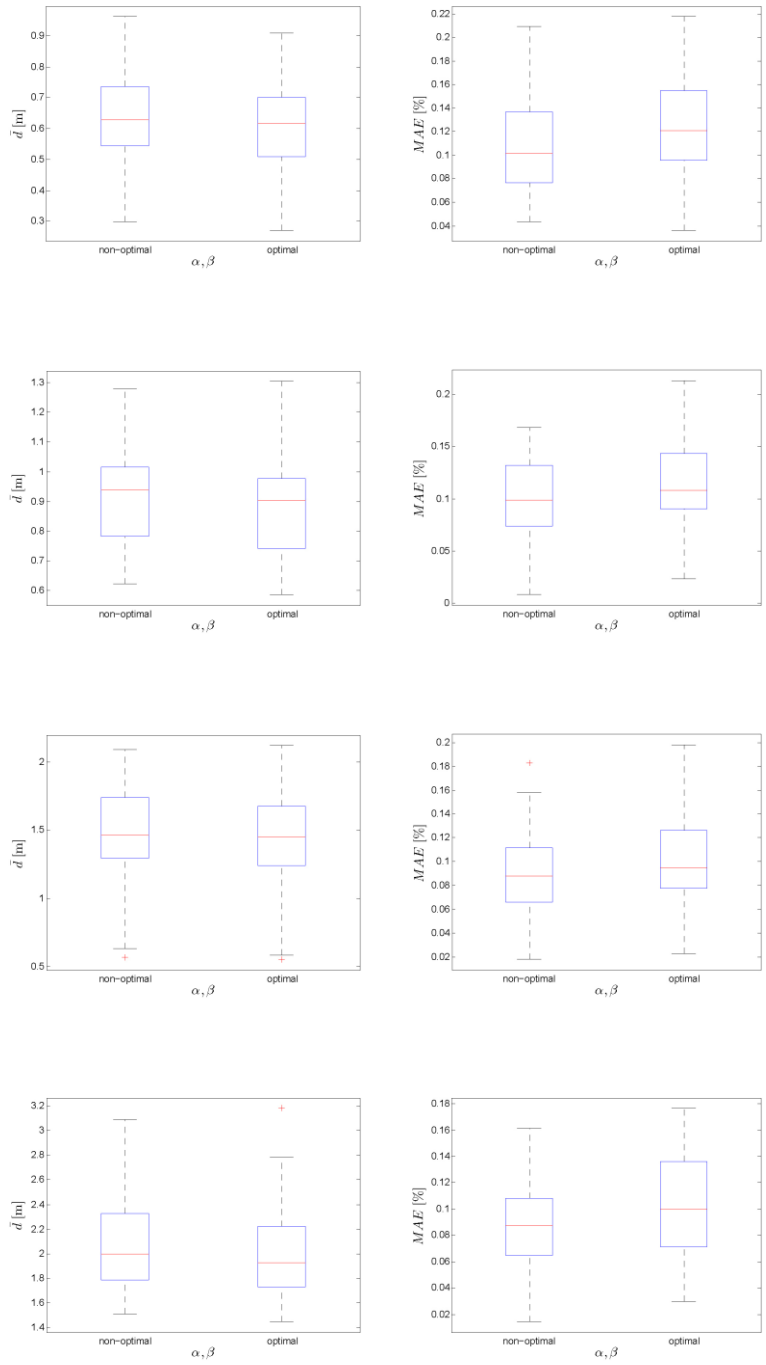

*(a) distance* *(b) MAE*

*Figure 9. Box-plot comparison of average distance (a) and MAE (b) for 40 robots. Non-optimal: α=β=1; optimal: α=2.65, β=2.55. Each box-plot comprises observations ranging from the first to the third quartile. The median is indicated by a horizontal bar, dividing the box into the upper and lower part. The whiskers extend to the farthest data points that are within 1.5 times the interquartile range. Outliers are shown with a plus symbol. The values were obtained from 50 experiments.*



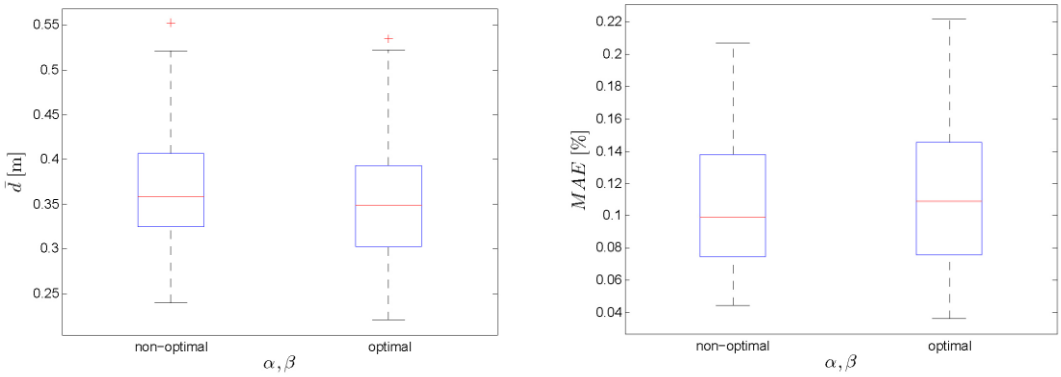

*(a) distance* *(b) MAE*

*Figure 10. Box-plot comparison of average distance (a) and MAE (b) for 100 robots. Non-optimal: α=β=1; optimal: α=2.65, β=2.55. Each box-plot comprises observations ranging from the first to the third quartile. The median is indicated by a horizontal bar, dividing the box into the upper and lower part. The whiskers extend to the farthest data points that are within 1.5 times the interquartile range. Outliers are shown with a plus symbol. The values were obtained from 50 experiments.*




*(a) distance* *(b) MAE*

*Figure 11. Box-plot comparison for 4 different robot arenas each involving 100 robots and 4 different randomly distributed targets: a) distance; b) MAE. Non-optimal: α=β=1; optimal: α=2.65, β=2.55. Each box-plot comprises observations ranging from the first to the third quartile. The median is indicated by a horizontal bar, dividing the box into the upper and lower part. The whiskers extend to the farthest data points that are within 1.5 times the interquartile range. Outliers are shown with a plus symbol. The values were obtained from 50 experiments.*




*(a) distance* *(b) MAE*

*Figure 12. Box-plot comparison for 100 robots and 4 random-valued, randomly distributed targets: a) distance; b) MAE. Non-optimal: α=β=1; optimal: α=2.65, β=2.55. Each box-plot comprises observations ranging from the first to the third quartile. The median is indicated by a horizontal bar, dividing the box into the upper and lower part. The whiskers extend to the farthest data points that are within 1.5 times the interquartile range. Outliers are shown with a plus symbol. The values were obtained from 50 experiments.*

The simulation setup 3 tests the swarms adaptability when the targets' location and targets' associated qualities are randomly chosen. The scenario considers a case of 100-robot swarm in search for 4 targets. The robot arena used in the simulations is 2.25m x 3.1875m dimension. As it can be noticed from the fig. 12, for the optimal set of control parameters' values the performance of the swarm of robots improved in terms of both deployment cost and distribution error.

## CONCLUSION AND FUTURE RESEARCH

Many applications of large multirobot systems require efficient task allocation in terms of individual and combined robots' utilities. The solution quality is analyzed using a defined performance metrics. In the presented study this was the mean absolute error of the robots' distribution in terms of the qualities of the available targets in the robot arena. In case of large, autonomous, multirobot systems, the scalability and the ability to adapt to different environments are the features of great importance. The presented experimental and simulation results showed that the proposed Distributed Bees Algorithm (DBA) provides the robot swarm with scalability in terms of the number of robots and number of targets, but also with the adaptability to a non-uniform distribution of the targets' qualities. The importance of control parameters is that they provide a mechanism to adjust the robot swarm behavior according to the task at hand and the available resources. In this chapter, the values of control parameters were tuned to bias the resulting robots' distribution towards the more favorable targets or to reduce the deployment costs.

Swarm Intelligence is a useful tool for solving a number of real-world problems. The decentralized approach in the design of multi-agent systems offers many advantages such as greater autonomy, scalability, robustness, and adaptability to a dynamically changing environment. This line of research was carried out in the domain of computer science and engineering, but it supports applications that go beyond

that as the modeling of multi-agent systems can be carried over to the domains of biology, medicine, sociology, economy, business, etc. The processes in nature and many processes in human society show emergent properties that are the result of multiple interactions between large numbers of individuals. Various scientific disciplines use different approaches to describe this phenomenon. By defining the relation between the stochastic processes on a lower level and the organized complexity on the system's global level, the predictability of such systems could be improved. This would have a high impact in the above-mentioned application domains. Moreover, the research work on Swarm Intelligence models can provide important feedback for the study of the natural swarms from which they were inspired.

## REFERENCES


1. Jamshidi, M. (2009). *System of Systems Engineering - Innovations for the 21st Century*. JohnWiley & Sons, New York, NY, USA.
2. Dorigo M. & Sahin E. (2004). Swarm Robotics - Special Issue. *Autonomous Robots, 17*:111-113.
3. Trianni V., Nolfi S. & Dorigo M. (2008). Swarm Robotics, *Design, 4433*(31):163–191.
4. Swarm Intelligence and Swarm Robotics - SISR 2010: Special Issue on Swarm Robotics (*Neural Computing & Applications*, vol. 19, no. 6, September 2010).
5. Engelbrecht, A. P. (2005). *Fundamentals of Computational Swarm Intelligence*, John Wiley & Sons, New York, NY, USA.
6. Gazi, P., Jevtić, A., Andina, D. & Jamshidi M. (2010). A mechatronic system design case study: Control of a robotic swarm using networked contro algorithms. In Proceedings of *4th Annual IEEE Systems Conference (SysCon 2010)*, pp. 169-173.
7. Beni, G. & Wang, J. (1989). Swarm intelligence in cellular robotic systems. *Proceedings of the NATO Advanced Workshop on Robotics and Biological Systems*, Il Ciocco, Tuscany, Italy.
8. Dudek, G., Jenkin, M. & Milios, E. (2002). A taxonomy of multirobot systems. In T. Balch & L. Parker (Eds), *Robot Teams: From Diversity to Polymorphism*, A.K. Peters, Natick, Massachusetts, pp. 3–22.
9. Gerkey, B. P. & Matarić, M. J. (2004). A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. *The International Journal of Robotics Research*, *23*(9): 939–954.
10. Jevtić, A., Gazi, P., Andina, D. & Jamshidi, M. (2010). Building a swarm of robotic bees. *World Automation Congress (WAC 2010)*, pp. 1–6.
11. Rana, O. F. & Stout, K. (2000). What is scalability in multi-agent systems?. *Proceedings of the 4th International Conference on Autonomous Agents*, AGENTS '00, ACM, New York, NY, USA, pp. 56–63.
12. Matarić, M. J., Sukhatme, G. S. & Østergaard, E. H. (2003). Multi-robot task allocation in uncertain environments. *Autonomous Robots*, *14*(2-3):255–263.
13. Michael, N., Zavlanos, M. M., Kumar, V. & Pappas, G. J. (2008). Distributed multi-robot task assignment and formation control. In Proceedings of *IEEE International Conference on Robotics and Automation (ICRA 2008)*, pp. 128–133.
14. Dias, M. B., Zlot, R., Kalra, N. & Stentz, A. (2006). Market-based multirobot coordination: A survey and analysis. *Proceedings of the IEEE*, *94*(7):1257–1270.
15. Franchi, A., Freda, L., Oriolo, G. & Vendittelli, M. (2009). The sensor-based random graph method for cooperative robot exploration. *IEEE/ASME Transactions on Mechatronics*, *14*(2):163–175.
16. Burgard, W., Moors, M., Stachniss, C. & Schneider, F. E. (2005). Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, *21*(3):376–386.

17. Ray, A. K., Benavidez, P., Behera, L. & Jamshidi, M. (2009). Decentralized motion coordination for a formation of rovers. *IEEE Systems Journal*, *3*(3):369–381.
18. Joordens, M. A. & Jamshidi, M. (2010). Consensus control for a system of underwater swarm robots. *IEEE Systems Journal*, *4*(1):65–73.
19. Berman, S., Halasz, A., Hsieh, M. A. & Kumar, V. (2009). Optimized stochastic policies for task allocation in swarms of robots. *IEEE Transactions on Robotics*, *25*(4):927–937.
20. Schwager, M., McLurkin, J., Slotine, J.-J. & Rus, D. (2009). From theory to practice: Distributed coverage control experiments with groups of robots. In *Experimental Robotics, ser. Springer Tracts in Advanced Robotics*, O. Khatib, V. Kumar, and G. Pappas (Eds.), Springer Berlin/Heidelberg, *54*:127–136.
21. Bonabeau, E., Dorigo, M. & Theraulaz, G. (1999). *Swarm intelligence: from natural to artificial systems*. New York, NY, USA: Oxford University Press, Inc.
22. Camazine, S., Franks, N. R., Sneyd, J.,Bonabeau, E., Deneubourg, J.-L. & Theraulaz, G. (2001). *Self-Organization in Biological Systems*. Princeton, NJ, USA: Princeton University Press.
23. Groß, R., Nouyan, S, Bonani, M., Mondada, F. & Dorigo, M. (2008). Division of labour in self-organised groups. In Proceedings of *10th international conference on Simulation of Adaptive Behavior: From Animals to Animats, ser. SAB '08*. Berlin, Heidelberg: Springer-Verlag, pp. 426–436.
24. Berman, S., Halasz, A., Kumar, V. & Pratt, S. (2007). Bio-inspired group behaviors for the deployment of a swarm of robots to multiple destinations. In Proceedings of *2007 IEEE International Conference on Robotics and Automation*, pp. 2318–2323.
25. Labella, T. H., Dorigo, M. & Deneubourg, J.-L. (2006). Division of labor in a group of robots inspired by ants' foraging behavior. *ACM Transactions on Autonomous and Adaptive Systems*, *1*(1):4–25.
26. Campo, A. & Dorigo, M. (2007). Efficient multi-foraging in swarm robotics. In Proceedings of *9th European Conference on Advances in Artificial Life, ser. ECAL'07*. Berlin, Heidelberg: Springer-Verlag, pp. 696–705.
27. Lerman, K., Jones, C., Galstyan, A. & Matarić, M. J. (2006). Analysis of dynamic task allocation in multi-robot systems. *International Journal of Robotics Research*, *25*(3):225–241.
28. Crespi, V., Galstyan, A. & Lerman, K. (2008). Top-down vs bottom-up methodologies in multi-agent system design. *Autonomous Robots*, *24*(3):303–313.
29. Farinelli, A., Iocchi, L., Nardi, D. & Ziparo, V. A. (2006). Assignment of dynamically perceived tasks by token passing in multirobot systems. *Proceedings of the IEEE*, *94*(7):1271–1288.
30. Jevtić, A., Gutiérrez, A., Andina, D. & Jamshidi, M. (2011). Distributed Bees Algorithm for Task Allocation in Swarm of Robots. *IEEE Systems Journal*, *5*(3):1–9.
31. Gutiérrez, A., Campo, A., Dorigo, M., Amor, D., Magdalena, L. & Monasterio-Huelin, F. (2008). An Open Localization and Local Communication Embodied Sensor. *Sensors*, *8*(8):7545–7563.
32. Gutiérrez, A., Campo, A., Dorigo, M., Donate, J., Monasterio-Huelin, F. & Magdalena, L. (2009). Open E-puck Range and Bearing Miniaturized Board for Local Communication in Swarm Robotics. In Proceedings of the *IEEE International Conference on Robotics and Automation*, pp. 3111–3116.
33. Gutiérrez, A., Campo, A., Monasterio-Huelin, F., Magdalena L. & Dorigo, M. (2010). Collective decision-making based on social odometry. *Neural Computing and Applications*, *19*(6):807–823.
34. Jevtić, A. & Gutiérrez, Á. (2011). Distributed Bees Algorithm Parameters Optimization for a Cost Efficient Target Allocation in Swarms of Robots. *Sensors*, *11*(11):10880-10893.
35. Golberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley: Lebanon, IN, USA.