This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999538, IEEE Access

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Task Allocation into a Foraging Task with a Series of Subtasks in Swarm Robotic System

WONKI LEE[1], NEIL VAUGHAN[2], AND DAEEUN KIM[3]

[1]Samsung Electronics, 1, Samsung-ro, Giheung-gu, Yongin-si, Gyeonggi-do, 17113, South Korea
[2]Department of Computer Science, Chester University, Chester, UK.
[3]School of Electrical and Electronic Engineering, Yonsei University, 50 Yonsei-ro, Seodaemun-gu, Seoul, 03722, South Korea

Corresponding author: DaeEun Kim (e-mail: daeeun@ yonsei.ac.kr).

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2020R1A2B5B01002395).

ABSTRACT In swarm robotic systems, task allocation is a challenging problem aiming to decompose complex tasks into a series of subtasks. We propose a self-organizing method to allocate a swarm of robots to perform a foraging task consisting of sequentially dependent subtasks. The method regulates the proportion of robots to meet the task demands for given tasks. Our proposed method is based on the response threshold model, mapping the intensity of task demands to the probability of responding to candidate tasks depending on the response threshold. Each robot is suitable for all tasks but some robots have higher probability of taking certain tasks and lower probability of taking others. In our task allocation method, each robot updates its response threshold depending on the associated task demand as well as the number of neighbouring robots performing the task. It relies neither on a centralized mechanism nor on information exchange amongst robots. Repetitive and continuous task allocations lead to the desired task distribution at a swarm level. We also analyzed the mathematical convergence of the task distribution among a swarm of robots. We demonstrate that the method is effective and robust for a foraging task under various conditions on the number of robots, the number of tasks and the size of the arena. Our simulation results may support the hypothesis that social insects use a task allocation method to handle the foraging task required for a colony's survival.

INDEX TERMS foraging task, response threshold model, sequential tasks, task allocation

I. INTRODUCTION

TASK allocation is a challenging subject in swarm robotic systems [1]–[3]. Depending on the needs of the systems, task allocation involves decomposing a task into sequentially interdependent subtasks and allocating a group of robots to perform different subtasks in parallel. Tasks are simultaneously performed at different locations and the posterior subtasks should be processed after completing the prior subtasks in order to complete the overall task. To increase the overall performance at a swarm level, a task allocation method is needed for balancing the task demands of subtasks by adaptively regulating the number of robots assigned to each subtask.

The goal is to determine how the individuals are allocated to each subtask in order to maximize the overall system performance. This subject has been studied in both biology and swarm robotics. In nature, an individual agent generally has very limited perception and knowledge of the environment, and there is no centralized control that guides the cooperation behaviors among a swarm of agents [4], [5]. Despite limited abilities, a swarm of agents in nature demonstrate effective task allocation by interacting with each other and sometimes perform complex tasks beyond the capability of a single agent [4]–[6].

Swarm robotic systems are primarily inspired by several social insects based on swarm intelligence. They are typically composed of homogeneous or heterogeneous robots. Traditionally, task allocation has used a centralized method [7]–[9]. However, depending on the types of tasks, it may not be possible for a controller to obtain all the necessary information, such as positions of robots and the current tasks assigned to all robots, in order to allocate tasks to each

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999538, IEEE Access

robot. Nevertheless, the goal requires to not only obtain the desired performance but also incorporate robustness and fault tolerance. In many applications, no communication or only local communication is available and knowledge of the environment is also limited. The individual robot must use only its local information to coordinate its tasks. Hence, the distributed method with no centralized controller has received attention. Currently, swarm intelligence and collective robotics have been inspired by adaptive behaviors observed in nature [10]–[13].

Many swarm robotic systems handle tasks consisting of a series of subtasks based on the task demand and there is an issue with how the robots in a swarm are allocated to several subtasks. Much research has focused on determining whether a robot performs a given task or not and on maximizing the colony-level performance [14], [15]. There is also a decision making approach with consensus of members in contrast to the task allocation of multiple subtasks. Consensus behaviors include coordinated motion [16], [17]. Individuals aggregate and move together in large groups that behave as if they were single organisms. Based on this behavior, autonomous formations of robot teams to perform more complex multi-robot tasks have recently attracted academic attention. Some studies have attempted to use multiple heterogeneous robots for coordinated emergency response and forming robot groups to undertake a number of tasks in disaster scenarios [3], [18]–[20].

A group of autonomous robots can be used to perform the whole task with a distributed approach [14], [15]. Task allocation can be regulated by measuring the waiting time between subtasks and probabilistically changing from the current subtask to another. Multiple heterogeneous robots can form teams to take advantage of synergies among robots and greedily select the task with the highest payoff measure [21].

Many types of sequential subtasks can be observed in social insects and various task allocation methods are employed to handle the tasks required for a colony's survival [22]–[24]. Various forms of task allocation strategies are observed in nature. In a direct transfer method, an item is transferred directly between workers. Collector bees take water or honey from the nectar source and transfer it directly to the resource-storing honeybees [4], [5], [25]. Efficient task allocation has been developed with the direct transfer [25]–[27]. Another strategy is available for cooperation of workers. Each worker can deliver goods to or take goods from a fixed or temporarily changing transfer location. There is no physical contact between individuals with an indirect transfer in foraging. An example of a fixed transfer area is a chamber with a pile of materials such as food or garbage which should be transferred into the nest or out of the nest. This transfer location can be located somewhere on the route between the source of materials and the nest. An example of the indirect transfer method can also be found in the termite, *Hodotermes mossambicus*. One group cuts grass and leaves into pieces and drops them on the ground. Then another group collects the pieces and takes them to their nest [6], [28]. This strategy can also be found in waste disposal processing. They create a garbage heap in their nest and this area is used as a cache for transfer. Some workers bring a piece of garbage and drop it near the garbage heap. Then other workers in the area remove the garbage from the nest [29].

An interesting example of multiple transfers is found in ants involving leaf fragment transfers [30]. There are three types of tasks among ants; arboreal cutters, cache exploiters, and carriers. Arboreal cutters ascend a tree to cut leaves and drop the leaf fragments to the ground. The pile of fallen leaves acts as a transfer location for task allocation. Cache exploiters then find the pile caches, cut the leaves into pieces and take them to the foraging trail, where they transfer them to carrier ants that transport them to the nest.

The foraging task is one of popular subjects used to demonstrate task allocation in swarm robotic systems. A group of robots are assigned to search for food items and deliver them from a resource area to specific locations. Jones and Mataric [31] proposed a task allocation method for swarm robots over a set of dynamically evolving concurrent tasks. It is achieved in a distributed manner by monitoring the ratio of task activities to the number of robots, and robots select tasks to be performed. A stochastic decision based on the response threshold model inspired by the behavior of insect societies assigns given tasks to individual robots in a swarm [10]. There have been various response threshold models for flexible task changes for the individual robot in dynamic environment [32]–[34].

A few studies address the problem of task allocation for sequentially interdependent subtasks. Pini et al. [14] proposed a task allocation method for a swarm of foraging robots by using a cost function, conceptually similar to the problem presented in this article. The robots autonomously calculate their traveled distance and deposit objects at an appropriate site. Then the robots performing a random walk can find objects and transfer them to the nest. The transportation of the object is executed as a sequence of subtasks performed by multiple robots, where task allocation improves the performance. Brutschy et al. [15] presented a method that objects are transferred directly between robots and any task change in the method is based on the interface delay (waiting time) experienced by the robots in one subtask relative to the interface delays experienced in another subtask. Zahadat and Schmickl [35] presented an adaptable task allocation of autonomous underwater robots based on social inhibition. A robot selects its own task based on the relative demand for a set of multiple tasks and response threshold values regulated by local interactions. Task allocation can also be applied to a group of heterogeneous robots [36], [37].

In this article, we tackle a foraging task consisting of a series of subtasks, and present an adaptive task allocation method based on the response threshold model [38], [39]. This model is composed of two factors; demand or stimulus of a given task and the response threshold for the corresponding task. The demand of a task decreases as an agent

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999538, IEEE Access

performs the task and the individuals with higher thresholds are unlikely to perform the task because the individuals with lower thresholds can handle the demand. However, if agents with lower thresholds lose the chance to perform the task, the demand of the task will increase and so agents with higher thresholds may be assigned to the task. Repetitive and continuous task allocations can produce the desired task distribution for a group of agents.

In order to achieve the optimal task allocation, robots in a swarm should change current subtasks if necessary. Two different levels of modeling are available; microscopic and macroscopic view. Microscopic modeling focuses on the individual robot and on interactions among the robots. It is difficult to apply mathematical analysis to the microscopic view. Yet local information can guide the global behavior for a swarm of agents. Previouly, a response threshold model and the threshold learning have been applied to colored puck collecting behavior of swarm robots [40]. Following the classical response threshold method, the task stimuli and the thresholds are defined such that a response threshold of each robot converges to the equilibrium state and this leads to the desired state of task allocation. With the convergence of the task allocation, the rate of agents working on tasks can come close to the global task demands. Since each agent has no information about the global task demand, they use the history queue for task demands and the proportion of neighboring agents for each task to estimate the global information.

Macroscopic modeling handles the swarm as a single system. The system of individual robots can be abstracted to a linear differential equation model [41], [42]. We provide the mathematical convergence of the task distribution in a swarm to show that instantaneous information of the task demands and the number of task workers in the local vicinity without any history queue can be sufficient to achieve efficient task allocation even for a series of subtasks. In addition, we demonstrate a self-organizing task allocation in the foraging task consisting of sequentially dependent subtasks. This work involves decomposing a task into sequentially inter-dependent subtasks and allocating a group of robots to the subtasks. The method handles an indirect transfer of food objects in the foraging task. A swarm of robots interact in the cache transfer area to determine their task allocation. Task allocation with our approach is supposed to adjust the task transition rate from one task to another in a macroscopic view, and for the purpose, the response threshold model for each robot controls the task change locally. There have been many task allocation methods with the response threshold model, but the methods often focus on the division of labor with appropriate cost function.

The paper is structured as follows. In Section II, we describe our task allocation problem and introduce the method that we propose for solving this problem. In Section III, we discuss the stability and convergence properties of the suggested method. In Section IV, we present the simulation environment for agent behavior consisting of a series of subtasks, and we report the corresponding results in Section V. Finally, in Section VI, we conclude the paper and discuss some possible future research directions.

II. PROBLEM DESCRIPTION AND MODELING

In this section, we describe the problem of task allocation for a foraging task consisting of sequential subtasks. The purpose of task allocation is to determine the optimal allocation of robots to subtasks to complete each transfer task as quickly as possible. If all the tasks take the same amount of time to perform, one possible solution is to adjust the population of robots performing specific tasks in proportion to the amount of the demand of the task. This involves balancing the number of robots performing each task and maximizing the swarm-level performance, that is, the number of food objects retrieved per time unit. Robots tend to specialize in specific tasks and it minimizes the number of task changes needed to produce the desired task distribution among robots. In our foraging task, robots obtain items from the resource and collected items are transported indirectly via a cache area to the nest.

A. DESCRIPTION OF FORAGING TASK

We assume that a foraging task is composed of two interdependent subtasks: resource-harvesting and resource-storing tasks. The simplified state diagram of robots performing the foraging task is shown in Fig. 1. Two primary behaviors will be performed.

Robots harvest from resources and deposit food in a cache area. Food from the cache area should be stored in a central nest, which is accomplished by other robots that transport food items to the nest. Harvesting robots travel to a resource area, pick up a food pellet and deliver it so that it can be processed later by a storing robot. To facilitate the transfer of food items to the nest, harvesting robots deposit their food items in a cache. Storing robots travel to a cache area, pick up a food pellet if one is present and deliver it to the central nest. The set of sequential tasks (harvesting from a specific resource area and storing food obtained from the cache in the central nest) are inter-linked. The harvesting task is a prior condition to the resource-storing task. The entire foraging task (that transfers one object from the resource area to the nest), is completed if both subtasks are completed. According to the multi-robot task allocation taxonomies proposed by Gerkey and Mataric [43], the categorization of the current work is given as single-task robots (ST), multi-robot tasks (MR), and instantaneous assignment (IA) problem.

The performance of the swarm e.g., the total number of objects stored in the central nest, relies on the number of robots in each area working on the two different subtasks and how the robots interact for object transfer. Varying the proportion of robots assigned to each subtask yields different performances at the overall swarm level. The subtask for storing robots is greatly affected by the subtask for harvesting robots, because food items should be transported by harvesting robots before they are stored in the nest. As mentioned

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999538, IEEE Access

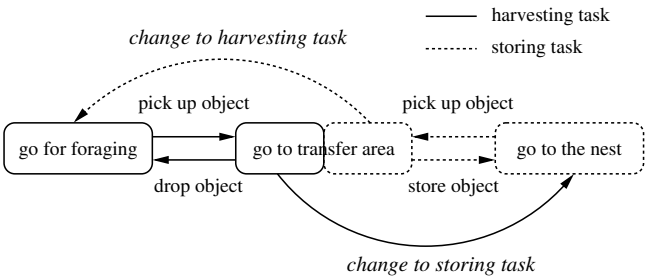


FIGURE 1: Simplified state diagram for robots performing the foraging task; solid line belongs to the harvesting subtask and dashed line belongs to the storing subtask. The behavior of each robot is determined by the subtask it is currently performing.

earlier, we consider the foraging task in which object transfer at the cache area as indirect. A robot that arrives at the cache area (transfer area) drops its food object and returns to the resource area. To obtain an optimal allocation, the number of robots working on the storing subtask should be regulated appropriately to transport objects located in the transfer area to the nest. The proposed method relies on balancing the number of objects located in the transfer area and the number of robots working on the storing task adaptively. The total performance also depends on the characteristics of the subtasks and/or the information about the surrounding environment.

B. PROPOSED METHOD

To solve the task allocation problem, we focus on adjusting the proportion of robots performing each subtask adaptively. If it is assumed that all the subtasks need the same amount of time to complete, one possible solution is to maintain the fraction of subgroup members equal to the proportion of subtasks to be done. We presume that the overall swarm performance is maximized when the allocation of robots to subtasks is optimal. We explain the linear differential model briefly, as presented by Halasz et al. [41] and Berman et al. [44]. Then we explain our proposed task allocation method for adjusting the proportion of robots performing each subtask adaptively, which is based on the response threshold model.

1) Modeling

We consider N robots and each robot can be allocated to one task among M tasks. We denote the number of robots performing task $i \in \{1, \ldots, M\}$ at time t by $n_i(t)$. Then the population fraction performing task i at time t is defined as $x_i(t) = n_i(t)/N$, and the vector of population fractions is represented by $x(t) = [x_1(t), \ldots, x_M(t)]^T$. The desired number of robots for task i is defined by $\bar{n}_i$ and the desired target distribution is the set of population fractions for each task, $\bar{x} = [\bar{x}_1, \ldots, \bar{x}_M]^T$, where $\bar{x}_i = \bar{n}_i/N$.

We can model the interconnection topology between M tasks via a directed graph, $G = (V, E)$. A set of vertices, V, represents M tasks. For a set of edges, E, tasks i and j are adjacent, (i, j), if a robot performing task i can change its task to task j. The graph G is strongly connected if a path exists between any task $i, j \in V$. To model the task transfer from one task to another, every edge in E is assigned a transition rate, $k_{ij}(t)$, where $k_{ij}(t)$ is the task transition probability per unit time for one robot previously executing task i to switch to task j. The transition rate from i to j does not equal the transition rate from task j to i, that is, $k_{ij}(t) \neq k_{ji}(t)$. Using the transition rate, the population fraction of robots executing task i is given by the linear equation [41], [42]:

$$\frac{dx_i(t)}{dt} = \sum_{\forall j|(j,i)\in E} k_{ji}(t)x_j(t) - \sum_{\forall j|(i,j)\in E} k_{ij}(t)x_i(t) \quad (1)$$

Assume the number of robots is conserved. Then Eq. (1) can be equivalently represented by the linear model to represent the average change rate of the population fractions executing their tasks.

$$\frac{dx(t)}{dt} = Kx(t) \quad (2)$$

Here, $K \in R^{M\times M}$ is a task transition matrix with the property

$$\sum_{\forall j|(i,j)\in E} K_{ij} = 0 \quad (3)$$

with $K_{ij} = k_{ij}$ for $i \neq j$ and $K_{ii} = 1 - \sum_{\forall j|(i,j)\in E} k_{ij}$. Eq. (2) is the formulation of Eq. (1) over all M tasks as a matrix equation, and it is referred to as the reaction rate equation. Using this model, the steady-state distribution of a group over various tasks can be controlled by appropriately selecting individual transition rates.

The studies of Halasz et al. [41] and Hsieh et al. [42] show that the system in Eq. (2) always converges to a unique task distribution regardless of the choice of K for a strongly connected graph. That is, it always achieves a unique stable equilibrium. With the desired distribution $\bar{x}$ with M tasks, the population can automatically distribute tasks accordingly among robots through the selection of the individual transition rates, k_{ij}. They studied how to determine the set of constant transition rates that result in fast convergence and how to minimize task transitions at the equilibrium state. However, constant transition rates force continuous task switching at the equilibrium state and the optimal transition rates are calculated with the help of the centralized method.

In our work, an individual robot selects its task autonomously without any centralized control and the transition rates are regulated adaptively, depending on the environmental conditions including the number of robots, number of tasks and size of foraging arena.

2) Task Selection Method

Our objective is to allocate a group of robots to achieve a desired distribution among various tasks. We assume that

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999538, IEEE Access

every robot has knowledge of the task demands and neighboring robots within a sensing range, but does not know the optimal task transition rate. Each individual selects a suitable task locally using local information without communication among robots, with no clear picture of what is occurring at the overall system level. Each robot autonomously decides whether to change the current task or not. The individual changes the current task according to the task selection function. The robot has a changeable threshold for each task and each robot responds differently to the same stimulus, which can coordinate the foraging behavior effectively. As a result, a collection of each robot's response produces swarm-level performance in a self-organized manner.

To explain the mathematical model, we assume that each robot x has a response threshold $\theta_{ix}(t)$ for task i. Then an individual robot's willingness to perform task i per unit time is a stochastic term calculated by a sigmoid function based on the response threshold model. For each robot, the task selection function for calculating the probability of changing tasks is given by

$$P_{ix}(t) = \frac{1}{1 + e^{-\frac{1}{\tau}[d_{ix}(t) - \theta_{ix}(t)]}}, \quad \forall i = 1, \ldots, M \tag{4}$$

where the task demand of task i for robot x is given as $d_{ix}(t)$, the threshold value is $\theta_{ix}(t)$, and the control parameter is τ, which determines the slope of the task selection probability. Task demand $d_{ix}(t)$ is the number of robots required to perform task i and the detected task demand by robot x within a limited sensing range is used. For $d_{ix}(t) < \theta_{ix}(t)$, the probability of a given task is close to 0 and for $d_{ix}(t) > \theta_{ix}(t)$, this probability is close to 1. At $d_{ix}(t) = \theta_{ix}(t)$, $P_{ix}(t) = 0.5$. In the remainder of this paper, we consider the case $\tau = 1$ for simplicity, but similar results can be obtained for any τ ($\tau > 0$).

Each robot decides to change the current task or not, relying on the probability obtained from Eq. (4). A robot can change the currently assigned task to task j if the probability of task j is the largest among all tasks. The threshold $\theta_{ix}(t)$ decreases and the task selection probability for task i increases. If the task demand is relatively lower than the fraction of robots performing that task, its threshold increases. From the overall system viewpoint, the task transition rate, $k_{ij}(t)$, can be defined as

$$k_{ij}(t) = \frac{n_i^j(t)}{n_i(t)} \tag{5}$$

where $n_i(t)$ is the number of robots performing task i, and $n_i^j(t)$ represents the number of robots currently performing task i and approaching the maximum probability for task j. That is, those robots will change their task to task j. The value $k_{ij}(t)$ changes depending on the condition of each robot but will decrease as the overall system converges to the desired task distribution.

3) Threshold Regulation Method

To obtain an adaptive task distribution, the task selection probability for a specific task can be increased by lowering the threshold of that task, or can be decreased by increasing the threshold. This threshold-updating process results in the emergence of specialized robots who are more responsive to tasks with lower thresholds than other robots, and this tendency produces a gradual migration of task allocation.

In the response threshold model, the threshold is usually updated after performing a task [38], [39], [45]. Completing a task induces a decrease in the threshold of that task and an increase in the thresholds of the other tasks not performed. The more often a robot performs a specific task, the lower its response threshold to this type of task, and vice versa, which leads to the robot specializing in performing a specific task.

In our task allocation method, the individual robot updates its response threshold considering not only the associated task demand but also using local information from other robots, that is, the task state of other robots. This tendency can be represented as follows:

$$\theta_{ix}(t+1) = \theta_{ix}(t) - \eta \{d_{ix}(t) - n_{ix}(t)\} \tag{6}$$

where η is a learning rate as a strengthening or weakening factor to regulate the threshold over time and $n_{ix}(t)$ is the number of robots performing task i observed by robot x. $\theta_{ix}(t)$ is constrained to the interval $[\theta_{min}, \theta_{max}]$. In this paper, the range of the threshold is set to $[\theta_{min}, \theta_{max}] = [1, 50]$ for simple application and η is selected as 1 to change the tendency of individuals for a specific task 2% per unit time.

Equation (6) means that threshold $\theta_{ix}(t)$ for task i is regulated by the fraction of robots performing that task. If the task demand $d_{ix}(t)$ is above $n_{ix}(t)$, the threshold of the corresponding task is decreased and thus the task selection probability for that task is increased. We refer to such a tendency as task specialization and this is further accelerated until $d_{ix}(t)$ drops below $n_{ix}(t)$. If there are more task demands than the fraction of robots performing that task, then another robot increases its chance to work on the task by lowering its threshold and increasing the probability to perform the new task. From this repetitive process, each robot eventually has a tendency to perform one specific task and this specialization reduces task changes at the equilibrium state, thus maintaining the desired division of labor.

III. ANALYSIS

In this section, we show that our task allocation method described in Eq. (1) has a stable equilibrium point that satisfies the desired target distribution $\bar{x}_i$ for task i, for $i = 1, \ldots, M$ (M is the number of tasks). We argue that with the help of the system described by the above equation for $i = 1, \ldots, M$ for all $(i, j) \in E$, with the condition in Eq. (7), the response threshold updating rule in Eq. (6) and the task selection function in Eq. (4), the robot population will converge almost certainly to $\bar{x} = [\bar{x}_1, \ldots, \bar{x}_M]^T$.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999538, IEEE Access

We consider the stability of the equilibrium state for our proposed method. The balancing condition needed is

$$k_{ij}\bar{x}_i = k_{ji}\bar{x}_j, \quad \forall (i,j) \in E \tag{7}$$

Then, consider the following Lyapunov function given by

$$V = \sum_{i=1}^{M} \frac{\bar{x}_i}{2}\left(1 - \frac{x_i}{\bar{x}_i}\right)^2 \tag{8}$$

where the time derivative of Eq. (8) is

$$\begin{aligned}\frac{dV}{dt} &= \sum_{i=1}^{M} \frac{(x_i - \bar{x}_i)}{\bar{x}_i}\frac{dx_i}{dt} \\ &= \sum_{i=1}^{M} \frac{(x_i - \bar{x}_i)}{\bar{x}_i}\left(\sum_{\forall j|(j,i)\in E} k_{ji}x_j - \sum_{\forall j|(i,j)\in E} k_{ij}x_i\right)\end{aligned} \tag{9}$$

Then, from Eq. (7), the above equation can be changed to

$$\begin{aligned}\frac{dV}{dt} &= \sum_{i=1}^{M} \frac{(x_i - \bar{x}_i)}{\bar{x}_i}\left(\sum_{\forall j|(i,j)\in E} \frac{\bar{x}_i}{\bar{x}_j}k_{ij}x_j - \sum_{\forall j|(i,j)\in E} k_{ij}x_i\right) \\ &= \sum_{i=1}^{M}\sum_{\forall j|(i,j)\in E}\left(\frac{x_j}{\bar{x}_j} - \frac{x_i}{\bar{x}_i}\right)k_{ij}(x_i - \bar{x}_i) \\ &= \sum_{\forall j|(i,j)\in E}\left(\frac{x_j}{\bar{x}_j} - \frac{x_i}{\bar{x}_i}\right)\phi_{ij}\end{aligned} \tag{10}$$

where $\phi_{ij} = k_{ij}(x_i - \bar{x}_i)$.

We can set up the convergence condition such that ϕ_{ij} has an opposite sign to $(x_j/\bar{x}_j - x_i/\bar{x}_i)$. Thus, if $x_j/\bar{x}_j > x_i/\bar{x}_i$, then $\phi_{ij} < 0$ and similarly, if $x_j/\bar{x}_j < x_i/\bar{x}_i$, then $\phi_{ij} > 0$. That is,

$$\frac{dV}{dt} = \sum_{\forall j|(i,j)\in E}\left(\frac{x_j}{\bar{x}_j} - \frac{x_i}{\bar{x}_i}\right)\phi_{ij} < 0 \tag{11}$$

Thus, the time derivative of the Lyapunov function evaluates to negative. In addition, consider that when all $\phi_{ij} = 0$ or when $x_i/\bar{x}_i = x_j/\bar{x}_j$ for all i, j, the time derivative of the Lyapunov function is always non-positive, so the system converges almost certainly to the desired distribution, $\bar{x}_i$.

If the number of robots are totally conserved,

$$\sum_{i=1}^{M} x_i = 1 \tag{12}$$

and there are two types of tasks $(M = 2)$, then Eq. (10) always satisfies the following condition

$$\frac{dV}{dt} = \sum_{\forall j|(i,j)\in E}\left(\frac{1 - x_i}{1 - \bar{x}_i} - \frac{x_i}{\bar{x}_i}\right)k_{ij}(x_i - \bar{x}_i) < 0 \tag{13}$$

The task transition rate k_{ij} has a positive value, $k_{ij} > 0$, and if $(x_i - \bar{x}_i) > 0$, then $((1 - x_i)/(1 - \bar{x}_i) - x_i/\bar{x}_i) < 0$ because $(1 - x_i)/(1 - \bar{x}_i) < 1$ and $x_i/\bar{x}_i > 1$. Similarly, if $(x_i - \bar{x}_i) < 0$, then $((1 - x_i)/(1 - \bar{x}_i) - x_i/\bar{x}_i > 0)$.

If there are more than two types of tasks, $(M \geq 3)$, Eq. (10) might produce a positive value. However, if we assume that task assignment is regulated between the robots performing task i and the other robots not performing task i, then this case satisfies the convergence condition

$$\frac{dV}{dt} = \sum_{\forall j|(i,j)\in E}\left(\frac{1 - \sum_{k=1,k\neq i}^{M} x_k}{1 - \sum_{k=1,k\neq i}^{M} \bar{x}_k} - \frac{x_i}{\bar{x}_i}\right)k_{ij}(x_i - \bar{x}_i) < 0 \tag{14}$$

If more robots are assigned to task i than the desired number, some robots currently performing task i should switch to other tasks including task j. Then sequentially, it can be assumed that a task is regulated between two groups, those performing task j and those not performing task j. In this manner, Eq. (10) converges asymptotically to $\bar{x} = [\bar{x}_1, \ldots, \bar{x}_M]^T$.

In a given foraging task, each robot has no information about the desired target distribution, $\bar{x}_i$, and the current distribution, x_i, so they instead regulate the task distribution with the following assumption,

$$\frac{x_j}{\bar{x}_j} \simeq \frac{n_j/N}{d_j/\sum_{j=1}^{M} d_j} \propto \frac{n_j}{d_j} \tag{15}$$

where n_j is the number of robots performing task j and d_j is the demand of task j where the demand can be estimated by the number of observed objects. Equation (15) indicates that task demand, d_j, is instead used to estimate the desired distribution, $\bar{x}_j$, and the number of robots observed in the vicinity, n_j, is used to estimate the current distribution, x_j. Then Eq. (10) can be rewritten as

$$\begin{aligned}\frac{dV}{dt} &= \sum_{\forall j|(i,j)\in E}\left(\frac{x_j}{\bar{x}_j} - \frac{x_i}{\bar{x}_i}\right)k_{ij}\bar{x}_i\left(\frac{x_i}{\bar{x}_i} - 1\right) \\ &\simeq \sum_{\forall j|(i,j)\in E}\left(\frac{n_j}{d_j} - \frac{n_i}{d_i}\right)k_{ij}\frac{\bar{x}_i}{d_i}(n_i - d_i)\end{aligned} \tag{16}$$

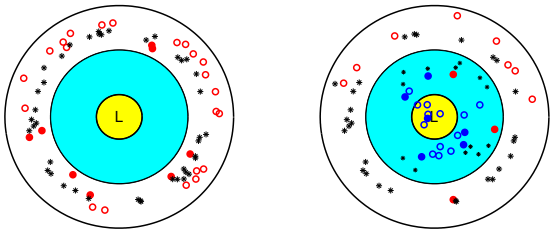
where $\bar{x}_i$ and d_i are non-negative values. If we design $k_{ij}(n_i - d_i)$ to have an opposite sign to $(n_j/d_j - n_i/d_i)$, by design if $n_j/d_j > n_i/d_i$, then $k_{ij}(n_i - d_i) < 0$; thus the transition rate k_{ij} has an opposite sign to $(n_i - d_i)$, and if $n_j/d_j < n_i/d_i$, then $k_{ij}(n_i - d_i) > 0$; the transition rate k_{ij} has the same sign as $(n_i - d_i)$, Thus Eq. (11) can converge to the desired distribution.

$$\frac{dV}{dt} \simeq \sum_{\forall j|(i,j)\in E}\left(\frac{n_j}{d_j} - \frac{n_i}{d_i}\right)k_{ij}\frac{\bar{x}_i}{d_i}(n_i - d_i) < 0 \tag{17}$$

The design rule obtained for converging to the desired performance is reflected in the threshold updating rule in Eq. (6). If the task demand $d_{ix}(t)$ of task i for robot x is relatively higher than the number of robots $n_{ix}(t)$ in the neighborhood, performing task i, it is necessary to increase the probability $P_{ix}(t)$ by decreasing the threshold $\theta_{ix}(t)$ for task i, which induces an increasing number of robots performing that task, so the transition rates k_{ij} and k_{ji} are regulated adaptively at the overall system level.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999538, IEEE Access



(a) (b)

FIGURE 2: Snapshot of simulation: (a) initial state where all robots are assigned to the harvesting task, and (b) desired state where the proper number of robots are assigned to each subtask, harvesting and storing, according to task demands. A swarm of robots are allocated to two subtasks, harvesting and storing tasks that are sequentially interdependent. Robots working on the harvesting task are represented by red circles and robots working on the storing task are represented by blue circles. Unladen robots moving to pick up objects are shown with empty circles and robots that transfer objects to their destination are shown with color-filled circles, respectively. The ground colors of the harvesting area (outer area), cache area (middle zone), and the central nest (inner circle) are white, sky-blue, and yellow, respectively. Food objects represented by star shapes are randomly scattered.

IV. SIMULATION ENVIRONMENT

In our foraging task scenario, the overall foraging task is partitioned into two subtasks; the harvesting subtask and the storing subtask. The harvesting and the storing subtasks have a sequential interdependency as they should be performed one after the other in order to complete the overall task sequence; transporting an object from the resource to the nest. We first explain the simulation environment. Then we evaluate the method using a swarm of robots in various simulations.

A. TASK DESCRIPTION

Fig. 2 shows a snapshot of the simulation. The environment is partitioned into three areas. We refer to the three areas marked with three different ground colors as the harvesting area, containing food items to be collected, the cache area (transfer area) as a common storage where objects are collected or dropped, and the central nest where objects are finally stored. The radius of the inner circle is 1 m while the middle radius and outer radius (that limits the range of the harvesting and storing tasks) are 3 m and 5 m, respectively. Additionally, a light source is located at the center of arena. All robots can sense the direction of the light source, which plays the role of a reference compass.

Each robot is equally capable of completing any subtask and only one type of subtask can be assigned to a robot at any time. A robot working in the harvesting area picks up an object and transports it to the cache area. By dropping the carrying object in the cache area, the object is transferred indirectly to robots working on the storing task. The robots working in the cache area collect objects only in that area and transport them to the nest. As objects are transferred indirectly, an additional behavior for object transfer between robots is not necessary.

Each robot moves around in the harvesting area, picks up one food pellet at a time, and delivers it to the cache (transfer) area. To transfer objects, robots head towards the light source until they arrive in the transfer area. They drop their object at random locations along the path to the light source and then go back to the harvesting area and move around to pick up another object. Robots working on the harvesting subtask shuttle between a food resource and the transfer area. Robots performing the storing task can pick up food pellets only within the transfer area and transport them to the central nest. In Fig. 2, robots working on the harvesting task are represented by red-colored circles and robots working on the storing task are represented by blue-colored circles. Unladen robots are shown with red or blue-colored empty circles and laden robots that transport food objects to their destination are shown with red or blue color-filled circles, respectively.

There are a fixed number of objects in the environment at any given time, given that they are simulated to appear and disappear, and no depletion occurs. The black star shapes represent objects. An object is picked up by a robot in the harvesting area and is transported to the central nest via the transfer area. A new object is generated in an arbitrary place in the harvesting area only after the object is dropped in the central nest, and the former object is simply cleared. The central nest has an infinite capacity for storage.

B. ROBOT BEHAVIORS

The simulation is implemented based on a Khepera-like robot model [46] using MATLAB. The robot has a round shape with a diameter of 12 cm. It can hold an object to be transported. The robot is equipped with infra-red sensors, used to perceive obstacles up to a distance of 20 cm and to sense the direction of a light source up to a distance of 16 m. Ground sensors positioned underneath the robot can detect the color of the floor ground. An omni-directional camera is mounted on top of a robot to perceive objects as well as neighboring robots up to a distance of approximately 3 m. Each robot can emit a light depending on what type of task it is currently performing, which helps to count the number of neighboring robots for each task. There is no communication between robots and the maximum moving speed of robots is 10 cm per time step.

Each robot performs a foraging task by collecting the closest object. It finds an object near itself and moves to grip the object. Robots can detect objects and neighboring robots by using their omni-directional camera. Simultaneously, a robot performs collision avoidance behaviors to prevent collisions with other robots or the outer wall of the arena. Each robot uses eight infrared sensors to estimate how close other robots or obstacles are positioned to them. The sensors are positioned uniformly around the robot to cover 180 degrees

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999538, IEEE Access

in front of the robot. A robot can change direction when the sensors detect any obstacle within sensing range.

Initially, all robots start at random positions in the harvesting area. When robot i with a food object arrives in the transfer area, it drops the object and updates its threshold θ_i. In our foraging task, a robot decides whether to change the current task or not, so only one threshold θ_i for the storing task is needed. For this, a robot handles the visual information from the omni-directional camera. The necessary information for threshold updating is a task demand d_i, which is estimated as the number of objects in the transfer area and the number of robots n_i performing the storing task. We assume that a robot can distinguish objects and robots from visual information obtained using an omni-directional camera mounted on top of a robot to perceive objects as well as neighboring robots up to a distance of about 3 m.

Each robot initially has the minimum threshold $\theta_i = \theta_{min}$ and updates it. Harvesting robots pick up a food pellet in the harvesting area and drop it in the cache area. If a robot's task selection probability becomes lower than that of the other tasks, they decide to switch to the other subtask. They stay in the cache area and start to perform the storing task from that moment. Here, there is no delay or extra cost involved in changing to a new task. Robots working on the storing task also decide whether to continue to perform the current task or change to the harvesting task based on local information. While robots perform the storing task, they update their thresholds every fixed period (every 20 time steps). Harvesting robots update their thresholds after they drop a food object in the cache area. To imitate the real robots with noise, maximum 10% random noise signals are added to sensor data and motor control for movement in simulation experiments. However, they never fail to obtain information from the camera image and picking or dropping behaviors never fail when transferring an object. These behaviors differ somewhat from real robots; therefore, the performance of real swarm robots may be slightly different from that of the simulated robots used in our experiments.

V. SIMULATION RESULTS

In the beginning of the foraging task, all robots are in the harvesting area (all robots are assigned to the harvesting task) and the number of food objects transported to the transfer area increases as time passes. Some robots with lower thresholds start switching to the storing task, and n_s (number of robots for the storing task) increases while n_h (for harvesting task) decreases. Then a smaller number of objects in the harvesting area will be delivered to the cache (transfer) area and an appropriate number of robots are eventually assigned to each subtask. We first show that the proposed task allocation method can regulate the labor by adjusting the number of robots performing the storing task properly depending on the changes in the number of food pellets in the cache area. Then we take various simulations to observe the performance of the proposed method by changing specific environmental conditions or robot parameters.

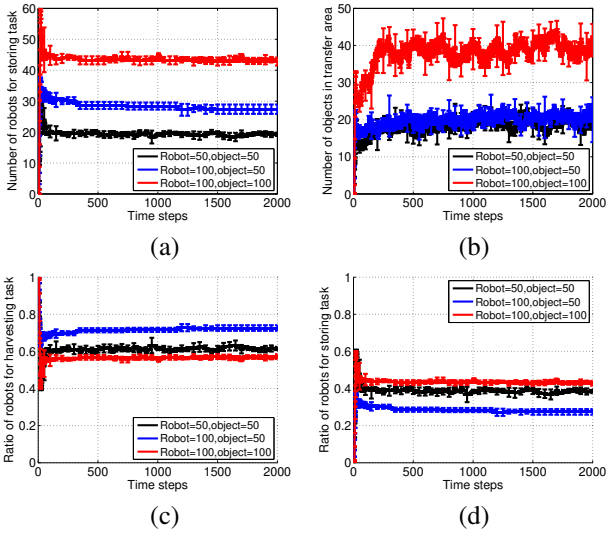


FIGURE 3: Various results from an original simulation, (a) change in the number of robots working on storing task, (b) change in the number of objects remaining in the transfer area, (c) progress of the proportion of robots assigned to harvesting task, and (d) progress of the proportion of robots assigned to storing task. As all robots are assigned to the harvesting task initially, they all start harvesting. Then the swarm is split properly by switching a proper number of robots from harvesting to storing.

In the dynamic task allocation environment, the amount of task demands or the number of robots may be changed, for example, by adding new objects or removing some robots. A robot may switch its task depending on the environmental situation. In a typical foraging task, the performance can be determined by the number of foraged objects, elapsed time, or consumed energy. If there is some cost associated with changing a task, it is also recommended to minimize task switching while maintaining the desired division of labor. In the simulations, we measure the number of objects located in the cache area, the number of robots working on the storing task, and the proportion of robots assigned to each subtask. Each simulation normally runs for 2,000 time steps in the test with 50 robots and 50 food objects, but it is changed depending on environmental conditions. The same number of robots as food items is used to compare the performance with the theoretical expectation (see Appendix A). Each simulation is repeated twenty times.

A. DIVISION OF LABOR

Fig. 3 shows the overall task allocation process of the proposed method (black line). The number of robots needed for the storing task is 20 in the test with 50 robots and 50 food objects, and the average number of objects remaining in the transfer area is also approximately 20 – see Fig. 3(a)-(b); similar to the theoretical estimation in Appendix A. One robot is needed to collect an object in the transfer area. It

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999538, IEEE Access

is expected that almost the same number of storing robots as the number of food objects in the transfer area will be maintained. For comparison, different numbers of robots and objects are considered (red and blue lines, respectively). Both values show the dynamic change of states with some variance as shown in Fig. 3(a)-(b), but the proportion of robots assigned to each task converges to a stable state as shown in Fig. 3(c)-(d). When the number of robots and objects are simultaneously doubled (red lines), similar proportions of robots are assigned to harvesting and storing tasks, as shown in Fig. 3(c)-(d), while the number of robots for the storing task and the number of objects in the transfer area are doubled, as shown in Fig. 3(a)-(b). However, when only the number of robots is doubled (blue lines), the objects in the transfer area remain at 20 and fewer robots compared to the total number of robots are needed to perform the storing task. Thus, the proportion of robots assigned to the harvesting task is higher.

In the foraging task, when the group harvesting the resource confronts a higher task demand at the transfer area by observing many food objects, some harvesting robots have a higher probability of switching to the storing subtask. When the storing group experiences a lower task demand, some robots storing objects will therefore be more likely to switch to the harvesting task. The number of objects remaining in the transfer area will gradually reach the equilibrium state. Additionally, the proportion of the two groups will become equal. There is some overshoot at an earlier time due to the tipping effect. This occurs because the initial thresholds of all robots are identical and improvements occur once individual thresholds begin to fluctuate.

In simulation experiments, all robots start with the minimum threshold (θ_{min}) and after updating the threshold, the distribution of thresholds changes. Each robot thus has a different threshold and this tendency induces a different response to the changes of environment and finally leads to the division of labor. A robot with lower thresholds has higher tendency to perform a specific task without task change (specializes at one task) and this leads to less frequent task changes for the individual robots. This type of task adaptability induces frequent task changes initially and fewer task changes are necessary after the task distribution converges to the stable state.

As shown in Fig. 4(a), during the first 200 simulation time steps, a steep increase is observed at the number of task changes, but after that time, the curve follows a gentle slope. The number of task changes becomes low as the task allocation converges to the desired stable state. Differences in the performances are evident if there are costs associated with task changes. In a constraint condition involving consumed energy for a task change, if the number of task changes increases, the total energy wasted in the foraging task increases rapidly. When there are more robots than objects (blue line), the number of task changes does not increase. A more detailed analysis about the effect of the ratio between the robots and objects will be future work. The performance of an

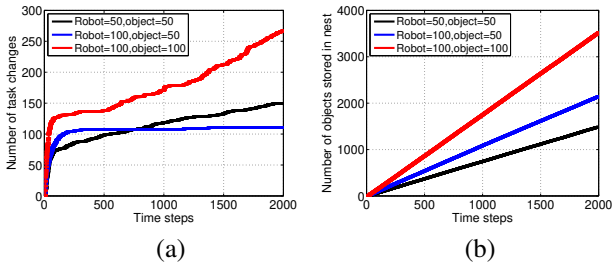


FIGURE 4: Results with varying numbers of robots and objects, (a) the total number of task changes of overall groups of robots, and (b) objects stored in nest.

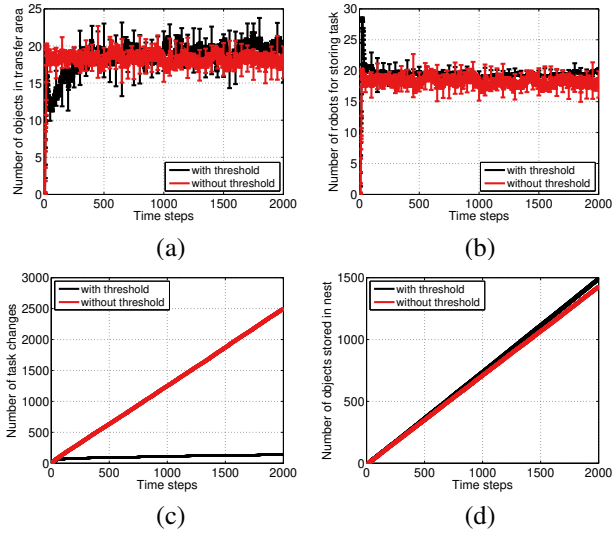


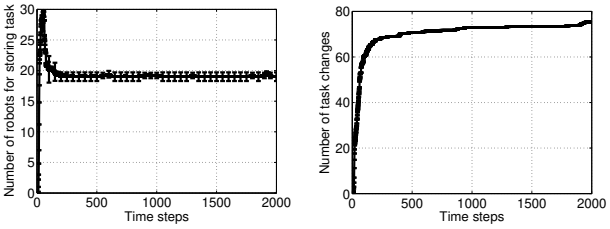
FIGURE 5: Comparison of the results with threshold and without threshold for the task selection function, (a) the number of objects in the transfer area, (b) the number of robots for the storing task, (c) the total number of task changes for the overall group, and (d) the total number of food objects stored in the nest.

overall system can be also evaluated by the number of objects stored in the nest. As shown in Fig. 4(b), the performance increases linearly without fluctuation despite task changes, indicating the task allocation method is effective. Improved performance is obtained by increasing the number of robots and objects. In our previous work [12], we show that an individual robot has to exert energy for wandering, gripping food objects, and task changes. The total number of wandering steps is almost constant and the energy required for performing foraging tasks is mainly dependent on the number of foraged objects and the number of task changes. Therefore, the difference in the performances among results is evident when there are differences in the number of task changes and gripping behaviors. If the cost of each behavior increases, the total energy wasted in the foraging task increases rapidly.

Fig. 5 shows the effect of using the response threshold

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999538, IEEE Access



(a) (b)

FIGURE 6: Results using only the number of objects for threshold regulation, but holding information of the desired target distribution, (a) the number of robots assigned to the storing task and (b) the total number of task changes.

model in the test with 50 robots and 50 food objects. Instead of using thresholds for task selection function (black line), each robot can just determine its task based on the instantaneous information (red line) for task i and robot x, which is given below:

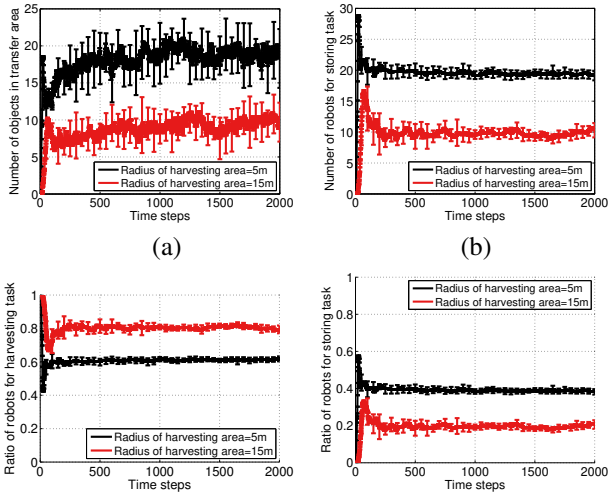
$$P_{ix}(t) = \frac{1}{1 + e^{-\frac{1}{\tau}[d_{ix}(t) - n_{ix}(t)]}}, \quad \forall i = 1, \ldots, M \quad (18)$$

where θ_{ix} in Eq. (4) with our approach is replaced by n_{ix} and n_{ix} is the number of robots performing task i observed by robot x. We can say that this approach is similar to the work found in Parker et al. [21]. As shown in Fig. 5(a), the number of objects in the transfer area reaches the desired level faster in the case without thresholds, but fewer robots with larger variations are assigned to the storing task and many more task changes are required even for similar conditions, as shown in Fig. 5(b)-(d). This implies that the method using the threshold θ_i benefits the decrease of the task switches. From the experiments, the work from Parker et al. [21] can have shortcomings of inducing frequent task changes, compared with our approach.

If the omni-directional camera has short-ranged detection, the performance may degrade, since only local area information can be covered to estimate the task demand or the current task workers. To confirm that the proposed response threshold model has a stable equilibrium point that satisfies the desired target distribution, we take another experiment, assuming each robot knows the desired number of robots remaining in the transfer area with an unlimited detective range of omni-directional camera as a special case. The desired number of robots is proportional to the size of the area in which they move around. Then the threshold can be updated depending only on the current number of robots in the transfer area as follows:

$$\theta_i(t+1) = \theta_i(t) - \eta\{\bar{n}_i - n_i(t)\} \quad (19)$$

Fig. 6 shows that the stable state is reached quickly and fewer task changes are needed using this approach under the same simulation conditions with Fig. 5. The number of robots for the storing task is the same as the result in Fig. 3(a). In Fig. 3(a) and Fig. 6(a), the number of robots performing



(a) (b) (c) (d)

FIGURE 7: Comparison of the results using different foraging area sizes, (a) number of objects remaining in the transfer area, (b) number of robots assigned to storing task, (c) proportion of robots assigned to harvesting task, (d) proportion of robots assigned to storing task.

storing tasks (n_s) is approximately 20, which is similar to the theoretical expectation as calculated by (see Appendix A)

$$n_s = \frac{r_s}{r_s + r_h} N \quad (20)$$

Then $n_s = 3/(3+5) \times 50 = 18.8$ for the case of 50 robots. From this result, we note that our proposed method nearly converges to the optimal task distribution.

B. EFFECT OF ARENA SIZE

We investigate the adaptability to the changes in the size of the foraging area. Fig. 7 shows comparison results to the previous simulations. In an original simulation, the radius of the harvesting area is 5 m (black line) and we change it to 15 m (red line). The number of robots performing storing tasks is approximately 20 when the radius of the harvesting area is 5 m. When the radius of the harvesting area is changed to 15 m, the number of storing robots is a little less than 10. This agrees with the expected results using Eq. (20); $n_s = 3/(3+15) \times 50 = 8.3$ when radius r_h is 15 m.

As the size of the foraging arena increases, robots need to move over longer distances to gather food objects and transport them to the transfer area, so the object-transfer rate is reduced and more robots are assigned to the foraging task. Then half as many robots are sufficient to perform the storing task to handle the lower number of food objects present in the transfer area. Depending on the environmental situation, an adaptive task allocation within the swarm group autonomously regulates the division of labor for sequential subtasks.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999538, IEEE Access

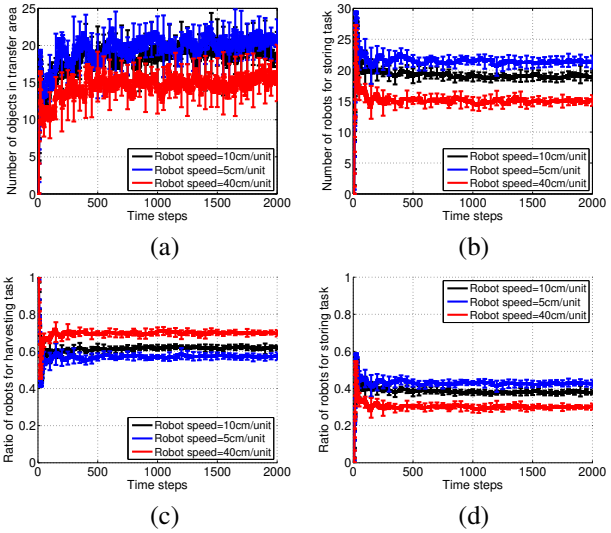


FIGURE 8: Comparison of the results with different moving speeds of robots performing storing task, (a) number of objects remaining in the transfer area, (b) number of robots assigned to storing task, (c) proportion of robots assigned to harvesting task, (d) proportion of robots assigned to storing task.

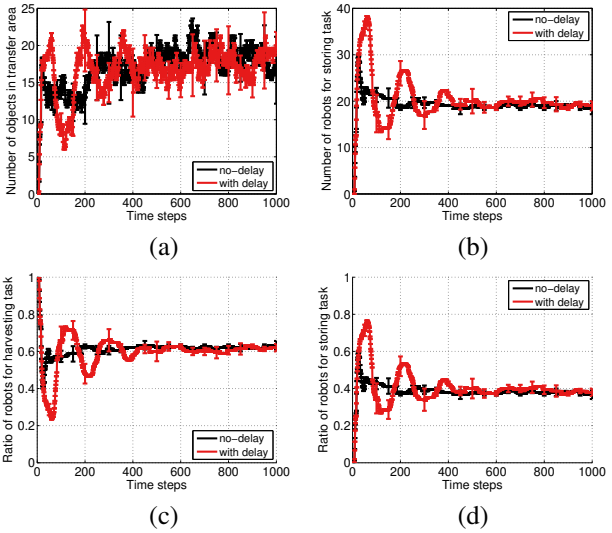


FIGURE 9: Comparison of the results with time delay and without delay for task changes, (a) number of objects remaining in the transfer area, (b) number of robots assigned to storing task, (c) proportion of robots assigned to harvesting task, (d) proportion of robots assigned to storing task.

C. EFFECT OF MOVING SPEED

In the next simulation, we investigate adaptability of the swarm to change in the moving speed of robots. To evaluate the difference in completing each subtask, we change the speed of robots performing the storing task while the speed of robots performing the harvesting task remains unchanged.

Fig. 8 presents the results. If the robot's speed is the same in the two groups, similar results (black line) are achieved to those shown in Fig. 3 and the proportion of robots assigned to the storing task converges to 40%. When we reduce the speed of robots working on the storing task by half (blue line), more robots are assigned to that task to reduce the number of objects collected in the cache transfer area. If the speed of robots performing the storing task is four times faster than robots performing the harvesting task (red line), fewer robots are needed to transport objects to the nest and the proportion of robots performing the storing task is reduced to 30%. In each case, the number of objects in the transfer area changes, as shown in Fig. 8(a) because our method in Eq. (6) depends on the number of robots working on the storing task. We confirm that our method dynamically allocates the task to a swarm of robots.

D. EFFECT OF DELAY FOR TASK CHANGE

In our simulations, we assume that there is no delay or extra cost to change tasks. However, in reality, some cost is incurred when changing a task; elapsed time, or consumed energy. Fig. 9 shows the comparison results when the robots pause at their current location for 50 simulation steps after changing their task. If extra time is required to change the task, the performances are overall worse. There are many overshoots, time is required for the robots to become stable, and higher numbers of task changes occur. However, the task distribution eventually converges to the previous results. The same number of objects remains in the transfer area and the same number of robots perform the storing task. The method remains effective even if the original (no-cost) assumption is changed.

E. EFFECT OF DISTURBANCE IN TASK ASSIGNMENT DISTRIBUTION

Finally, we investigate how robustly the system responds to an abrupt change in the task distribution. At time step 1,000, all robots assigned to the storing task are kidnapped from the swarm. The environment is the same as the original simulation. Fig. 10 presents the behaviors of the swarm in response to the change and shows that the swarm reacts properly by allocating a proper number of robots to the storing task. The task allocation among the remaining robots performing the harvesting task is regulated and the robots with low thresholds switch their current task to the storing task. Thus, it eventually reduces the task demand of the storing task as robots switch to it. A new proportion of robots are assigned to each task, and the proportion converges to a stable state within a short time.

VI. DISCUSSION

Comparing to prior works that generally update the response threshold after performing tasks, our method suggests that task allocation is regulated by the relative difference between the number of tasks not completed and the number of

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999538, IEEE Access

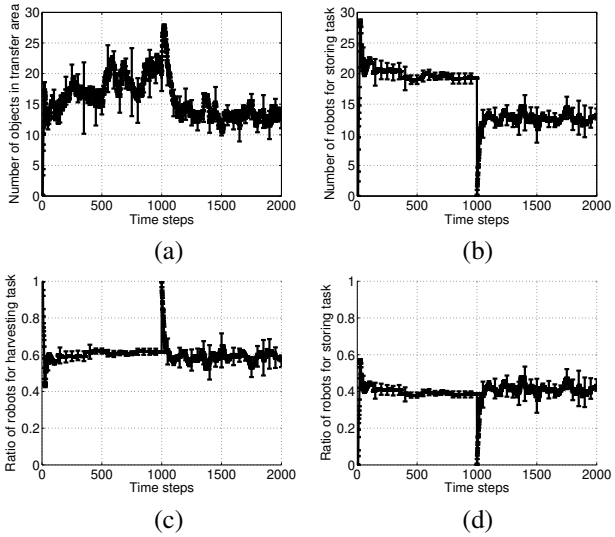


FIGURE 10: Results with sudden changes in the number of robots, (a) number of objects remaining in the transfer area, (b) number of robots assigned to the storing task, (c) proportion of robots assigned to the harvesting task, (d) proportion of robots assigned to the storing task

neighboring robots performing their corresponding subtasks. Performance of a task affects not only the task demand perceived by other robots but also the individual's preference with regard to that task. Task allocation can be efficiently organized by considering the internal state preference of the individual such as the task preference and also the external state obtained from the interaction with the environment. We also show the mathematical convergence of task distribution for the suggested approach. Our proposed method demonstrates a self-organizing process to allocate the whole task to a group of mobile robots, where the individual decisions of robots are based on the surrounding local information. Thus, it does not require a central controller and the robots do not require communication with each other. Each robot decides whether to switch between subtasks depending on the local information available. Repetitive and continuous task allocation leads to the desired task distribution at a group level. Especially for our foraging task, the system will approximately converge to the desired task distribution as demonstrated in the simulations.

The individual decision to switch between multiple subtasks depends on the response threshold, which differentiates responses to the same task demand, and the threshold value is updated by relying on a robot's individual perception. The simulation results show that a swarm using the proposed method is able to allocate its individuals to sequential subtasks adaptively and we can conclude that the method is adaptive as the robots successfully re-distribute the robots to subtasks even if the environmental conditions change. We will further study the efficiency with much more dynamic environments, such as the RoboCup domain. It would provide more convincing argument if this proposed method is indeed effective even in real dynamic environments.

VII. CONCLUSION

Task allocation in a swarm of robots involves decomposing a complex task into subtasks. In this paper, we present a foraging task in which a group of robots are supposed to collect food pellets and transfer them to the central nest. Harvesting robots transport them from the resource area to a common storage called the cache area (transfer area), and storing robots transport them from the common storage area to the central nest. We propose a dynamic task allocation method for regulating the fraction of robots proportionally to the fraction of task demands. Our contribution is to provide effective local task change of the individual robots with the response threshold model but without commuication, ultimately reaching the desired task distribution for a swarm, and also show the mathematical convergence of the task distribution. It is a self-organized method for decomposing a task into sequentially interdependent subtasks and allocating the individuals in a swarm to perform subtasks in parallel. This approach has the effect of reducing interference among the individual robots because different types of workers become more segregated and the improved transport efficiency allows for better overall swarm performance.

A further direction of work concerns the method for selecting task allocation strategies adaptively. *Atta cephalotes*, known as leaf-cutting ants, use various strategies to handle their materials such as food and garbage [29], [47]. When an ant finds a food source, it selects one of three strategies. The first strategy is to just return to the nest, carrying food by itself; in this case, the ant has no task allocation strategy. As another strategy, the forager ant directly passes the food to another ant. In the last strategy, it drops the food in the middle of the pheromone trail so that other ants can bring it to the nest by an indirect transfer. They freely change their strategies and this could be a possible extension of the work presented in this article. Furthermore, the idea can be applied to task transfer between heterogeneous robots. In addition, we wish to test experiments in the real world by building a swarm of real mobile robots.

APPENDIX A

If there are two types of foraging tasks in a circular arena, the desired task distribution can be calculated mathematically. To balance the workload among sequential subtasks in Euclidean space, the desired number of robots for each task is proportional to the number of objects in the given area and is inversely proportional to the area in which they move around if all robots are homogeneous in their ability for performing tasks. Then for a stable equilibrium point, the balancing condition is defined as

$$\frac{d_s}{r_s^2} n_s = \frac{d_h}{r_h^2} n_h \tag{21}$$

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999538, IEEE Access

where d_s and d_h are the number of objects in storing and harvesting areas with radius r_s and r_h, respectively. If the number of robots and objects are the same, Eq. (21) can be rewritten as

$$\frac{n_s}{r_s^2} n_s = \frac{(N - n_s)}{r_h^2} (N - n_s) \tag{22}$$

where N is the number of robots and objects. Then,

$$\frac{n_s^2}{r_s^2} = \frac{(N - n_s)^2}{r_h^2} \quad \rightarrow \quad \frac{n_s}{r_s} = \frac{(N - n_s)}{r_h} \tag{23}$$

Equation (23) means that the proportion of task allocation is decided according to the radius of the foraging area. Finally, the number robots performing storing task can be defined as

$$n_s = \frac{r_s}{r_s + r_h} N \tag{24}$$

REFERENCES

[1] D. Floreano, and L. Keller, "Evolution of adaptive behaviour in robots by means of Darwinian selection," PLoS biology, vol. 8, no. 1, pp. e1000292, 2010.

[2] L. Jin and S. Li, "Distributed Task Allocation of Multiple Robots: A Control Perspective", IEEE Trans. on Cybernetics, Vol. 49, No. 5, pp. 693–701, 2018.

[3] W. Zhao, Q. Meng, and P. Chung, "A Heuristic Distributed Task Allocation Method for Multivehicle Multitask Problems and Its Application to Search and Rescue Scenario", IEEE Trans. on Cybernetics, Vol. 46, No. 4, pp. 902–915, 2015.

[4] T. D. Seeley, "The wisdom of the hive: the social physiology ofhoney bee colonies," Cambridge, MA: Harvard UniversityPress, vol. 1995, 1995.

[5] J. H. Fewell and S. M. Bertram, "Division of labor in a dynamic environment: response by honeybees (apis mellifera) to graded changes in colony pollen stores," Behavioral ecology and sociobiology, vol. 46, no. 3, pp. 171–179, 1999.

[6] C. Anderson, J. J. Boomsma, and J. Bartholdi III, "Task partitioning in insect societies: bucket brigades," Insectes Sociaux, vol. 49, no. 2, pp. 171–180, 2002.

[7] L. E. Parker, "Alliance: An architecture for fault tolerant multirobot cooperation," Robotics and Automation, IEEE Transactions on, vol. 14, no. 2, pp. 220–240, 1998.

[8] M. B. Dias, R. Zlot, N. Kalra, and A. Stentz, "Market-based multirobot coordination: A survey and analysis," Proceedings of the IEEE, vol. 94, no. 7, pp. 1257–1270, 2006.

[9] E. Nunes and M. L. Gini, "Multi-robot auctions for allocation of tasks with temporal constraints." in AAAI, 2015, pp. 2110–2116.

[10] E. Bonabeau, M. Dorigo, and G. Theraulaz, Swarm intelligence: from natural to artificial systems. Oxford university press, 1999.

[11] D. Zhang, G. Xie, J. Yu, and L. Wang, "Adaptive task assignment for multiple mobile robots via swarm intelligence approach," Robotics and Autonomous Systems, vol. 55, no. 7, pp. 572–588, 2007.

[12] W. Lee and D. Kim, "History-based response threshold model for division of labor in multi-agent systems," Sensors, vol. 17, no. 6, p. 1232, 2017.

[13] M. Mavrovouniotis, C. Li, and S. Yang, "A survey of swarm intelligence for dynamic optimization: algorithms and applications," Swarm and Evolutionary Computation, 2017.

[14] G. Pini, A. Brutschy, C. Pinciroli, M. Dorigo, and M. Birattari, "Autonomous task partitioning in robot foraging: an approach based on cost estimation," Adaptive behavior, vol. 21, no. 2, pp. 118–136, 2013.

[15] A. Brutschy, G. Pini, C. Pinciroli, M. Birattari, and M. Dorigo, "Self-organized task allocation to sequentially interdependent tasks in swarm robotics," Autonomous agents and multi-agent systems, vol. 28, no. 1, pp. 101–125, 2014.

[16] O. Arslan, D. P. Guralnik, and D. E. Koditschek, "Coordinated robot navigation via hierarchical clustering," IEEE Transactions on Robotics, vol. 32, no. 2, pp. 352–371, 2016.

[17] W. Lee and D. Kim, "Autonomous shepherding behaviors of multiple target steering robots," Sensors, vol. 17, no. 12, p. 2729, 2017.

[18] P. Scerri, A. Farinelli, S. Okamoto, and M. Tambe, "Allocating tasks in extreme teams," in Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems. ACM, 2005, pp. 727–734.

[19] L. Vig and J. A. Adams, "Coalition formation: From software agents to robots," Journal of Intelligent and Robotic Systems, vol. 50, no. 1, pp. 85–118, 2007.

[20] S. D. Ramchurn, A. Farinelli, K. S. Macarthur, and N. R. Jennings, "Decentralized coordination in robocup rescue," The Computer Journal, vol. 53, no. 9, pp. 1447–1461, 2010.

[21] J. Parker, E. Nunes, J. Godoy, and M. Gini, "Exploiting spatial locality and heterogeneity of agents for search and rescue teamwork," Journal of Field Robotics, vol. 33, no. 7, pp. 877–900, 2016.

[22] F. L. Ratnieks and C. Anderson, "Task partitioning in insect societies," Insectes sociaux, vol. 46, no. 2, pp. 95–108, 1999.

[23] D. P. Mersch, A. Crespi, and L. Keller, "Tracking individuals shows spatial fidelity is a key regulator of ant social organization," Science, Vol. 340, No. 6136, pp. 1090–1093, 2013.

[24] E. Ferrante, A. E. Turgut, E. Duéñez-Guzmán, M. Dorigo, and T. Wenseleers, "Evolution of self-organized task specialization in robot swarms," PLoS computational biology, vol. 11, no. 8, p. e1004273, 2015.

[25] T. Schmickl and H. Hamann, "Beeclust: A swarm algorithm derived from honeybees," Bio-inspired Computing and Communication Networks. CRC Press (March 2011), 2011.

[26] D. Agrawal and I. Karsai, "The mechanisms of water exchange: the regulatory roles of multiple interactions in social wasps," PloS one, vol. 11, no. 1, p. e0145560, 2016.

[27] W. Lee and D. Kim, "Handling interference effects on foraging with bucket brigades," Bioinspiration & biomimetics, vol. 12, no. 6, p. 066001, 2017.

[28] R. H. Leuthold, O. Bruinsma, and A. v. Huis, "Optical and pheromonal orientation and memory for homing distance in the harvester termite hodotermes mossambicus (hagen)," Behavioral ecology and sociobiology, vol. 1, no. 2, pp. 127–139, 1976.

[29] A. G. Hart and F. L. Ratnieks, "Task partitioning, division of labour and nest compartmentalisation collectively isolate hazardous waste in the leafcutting ant atta cephalotes," Behavioral Ecology and Sociobiology, vol. 49, no. 5, pp. 387–392, 2001.

[30] H. G. Fowler and S. Robinson, "Foraging by atta sexdens (formicidae: Attini): seasonal patterns, caste and efficiency," Ecological Entomology, vol. 4, no. 3, pp. 239–247, 1979.

[31] K. Lerman, C. Jones, A. Galstyan, and M. J. Matarić, "Analysis of dynamic task allocation in multi-robot systems," The International Journal of Robotics Research, vol. 25, no. 3, pp. 225–241, 2006.

[32] M. J. Krieger, J.-B. Billeter, and L. Keller, "Ant-like task allocation and recruitment in cooperative robots," Nature, vol. 406, no. 6799, pp. 992–995, 2000.

[33] T. H. Labella, M. Dorigo, and J.-L. Deneubourg, "Division of labor in a group of robots inspired by ants' foraging behavior," ACM Transactions on Autonomous and Adaptive Systems (TAAS), vol. 1, no. 1, pp. 4–25, 2006.

[34] M. Castillo-Cagigal, E. Matallanas, I. Navarro, E. CaamaÃśo-MartÃ■n, F. Monasterio-Huelin, and A. Gutierrez, "Variable Threshold Algorithm for Division of Labor Analyzed as a Dynamical System", IEEE Trans. on Cybernetics, Vol. 44, No. 12, pp. 2242–2252, 2014.

[35] P. Zahadat and T. Schmickl, "Division of labor in a swarm of autonomous underwater robots by improved partitioning social inhibition," Adaptive Behavior, vol. 24, no. 2, pp. 87–101, 2016.

[36] N. Elsayed and K. Al-Wahedi, "Utilizing task partitioning for self-organized allocation of partially sequential tasks," in Proc. of IEEE Int. Conf. on Systems, Man, and Cybernetics, pp. 3030–3035, 2015.

[37] W. Hu and L. Liu, "Cooperative Output Regulation of Heterogeneous Linear Multi-Agent Systems by Event-Triggered Control", IEEE Trans. on Cybernetics, Vol. 47, No. 1, pp. 105–116, 2016.

[38] E. Bonabeau, A. Sobkowski, G. Theraulaz, and J.-L. Deneubourg, "Adaptive task allocation inspired by a model of division of labor in social insects." in BCEC, 1997, pp. 36–45.

[39] G. Theraulaz, E. Bonabeau, and J. Denuebourg, "Response threshold reinforcements and division of labour in insect societies," Proceedings of the Royal Society of London. Series B: Biological Sciences, vol. 265, no. 1393, pp. 327–332, 1998.

[40] W. Lee and D. Kim, "Adaptive approach to regulate task distribution in swarm robotic systems," Swarm and evolutionary computation, vol. 44, no. 12, p. 1108–1118, 2019.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2999538, IEEE Access

[41] A. Halász, M. A. Hsieh, S. Berman, and V. Kumar, "Dynamic redistribution of a swarm of robots among multiple sites," in 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2007, pp. 2320–2325.
[42] M. A. Hsieh, Á. Halász, S. Berman, and V. Kumar, "Biologically inspired redistribution of a swarm of robots among multiple sites," Swarm Intelligence, vol. 2, no. 2-4, pp. 121–141, 2008.
[43] B. P. Gerkey and M. J. Matarić, "A formal analysis and taxonomy of task allocation in multi-robot systems," The International Journal of Robotics Research, vol. 23, no. 9, pp. 939–954, 2004.
[44] S. Berman, Á. Halász, M. A. Hsieh, and V. Kumar, "Optimized stochastic policies for task allocation in swarms of robots," IEEE Transactions on Robotics, vol. 25, no. 4, pp. 927–937, 2009.
[45] V. A. Cicirello and S. F. Smith, "Wasp-like agents for distributed factory coordination," Autonomous Agents and Multi-agent systems, vol. 8, no. 3, pp. 237–266, 2004.
[46] F. Mondada, E. Franzi, and P. Ienne, "Mobile robot miniaturisation: A tool for investigation in control algorithms," Experimental robotics III, pp. 501–513, 1994.
[47] J. Cherrett, "The foraging behaviour of atta cephalotes l. (hymenoptera, formicidae)," The Journal of Animal Ecology, pp. 387–403, 1968.

WONKI LEE received his B.E.,M.S. and Ph.D degree from the Department of Electrical and Electronic Engineering of Yonsei University. Currently he is a staff engineer in Samsung electronics. His research interests are in the areas of visual navigation, artificial intelligence, neural networks.

NEIL VAUGHAN is an associate professor at University of Exeter, and he was a research fellow of the Royal Academy of Engineering since 2018. He was a senior lecturer in Computer Science at University of Chester. He is a Senior Member of Artificial Intelligence and Simulation of Behaviour (AISB), Senior member of IEEE, Computational Intelligence Society, Editor of the Journal of Behavioural Robotics.

DAEEUN KIM received his B.E. and M.S. from the Department of Computer Science and Engineering of Seoul National University and the University of Michigan at Ann Arbor, respectively. He received his Ph.D. degree from the University of Edinburgh in 2002. From 2002 to 2006, he was a research scientist in Max Planck Institute for Human Cognitive and Brain Sciences. Currently he is a professor in Yonsei University in Corea. His research interests are in the areas of biorobotics, autonomous robots, artificial life, neural networks, and neuroethology.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/.