AY2020 Master's Thesis

Behavioral Strategies for Energy-Aware Multi-Agent Continuous Cooperative Patrolling Problems subject to Quality Requirement

WU Ling-ying

A Thesis Submitted to the Department of Computer Science and Communications Engineering, the Graduate School of Fundamental Science and Engineering of Waseda University in Partial Fulfillment of the Requirements for the Degree of Master of Engineering

Student ID	5118FG18-6
Date of Submission	July 18th, 2020
Advisor	Prof. Toshiharu Sugawara
Research Guidance	Research on Intelligent Software

Contents

Abstract

This paper proposes a method of autonomous strategy learning for multiple cooperative agents integrated with a series of behavioral strategies aiming at reduction of energy cost on the premise of satisfying the quality requirement in continuous patrolling problems. The research of the multi-agent reinforcement learning and patrolling problem has been widely conducted from different aspects. However, the issue of energy minimization has not been sufficiently studied. When considering real-world applications with a trade-off between energy efficiency and level of perfection, it is usually more desirable to minimize the energy cost and carry out the tasks to the required level of quality instead of fulfilling tasks perfectly by ignoring energy efficiency. We present a series of coordinated behavioral strategies and an autonomous learning method of target decision strategies to reduce energy consumption subject to the given quality requirement in multi-agent continuous patrolling problems. Our previous method of target decision strategy learning is extended by incorporating a number of behavioral strategies, with which agents individually estimate whether the requirement is reached and then modify their action plans to save energy. In addition, we improved our algorithm of requirement estimation to avoid concentration of agents since they are given the knowledge of the work environment in advance. The experimental results show that our proposal enables the agents to learn to select appropriate behavioral planning strategies according to performance efficiency and energy cost. Then, they individually estimate whether the given requirement is reached and modify their action plans to save energy. Furthermore, we found that agents with the advanced requirement estimation method could achieve fair patrolling by introducing local observations.

1 Introduction

Along with the rapid growth in robotics and computational technologies, robot applications have gained popularity in various real-world environments. Not only performing real-world tasks in a human-like manner but also working in places and situations that are difficult for humans, intelligent agents have shown great potentialities of offering different types of functionality and transcending human performance. However, restrictions of speed, movement, and battery capacity limit the performance of single-robot systems. To break this limit, multi-robot systems are expected to complete tasks by compensation and cooperation, as well as to be fault tolerant.

A multi-agent system (Wooldridge, 2009) is an extension of the agent technology where several loosely connected autonomous agents act in an environment to achieve a common goal. The interaction between agents can either be cooperative or selfish. Our study pays attention to the cooperative case, where the agents attempt to jointly solve tasks or to maximize utility through their interaction (Panait, 2005). More importantly, coordination between multiple agents on their decision-making is crucial for them to achieve the optimal performance of the group as a whole in complex and large scale tasks (Almeida et al., 2004).

There are a wide range of real-world applications benefiting from the study of multi-agent systems, including cloud computing, market simulation, system diagnosis, and monitoring (Xie and Liu, 2017). In this study, we tackle the *continuous cooperative patrolling problem* (CCPP) addressed by Sugiyama et al. (Sugiyama et al., 2016), in which an agent is an intelligent program capable of autonomously deciding its action plan and control a portable robot to continuously move around a given area and visit locations with required and different frequencies for given purposes. The multi-agent CCPP is the abstract problem for many real-world applications that are complex and require appropriate coordination and cooperation between agents, such as cleaning, security, and surveillance patrolling tasks. A key constraint on the multi-agent environment is that agents may not at any given time know everything about the world that other agents know. Also, agents may only have CPUs with limited performance, a constrained amount of battery, and limited battery capacities. Accordingly, it is reasonable to propose two important assumptions in the multi-agent CCPP, *shallow coordination with restricted communication* and *periodical battery recharging*.

Multi-agent system problem has spawned increasing interest in real-world applications. It is pointed out that research in the multi-agent patrolling problem field should be oriented toward solutions with applicability in the real-world (Portugal and Rocha, 2013). Realistic scenarios must be considered when deploying actual systems (Iocchi et al., 2011). Nowadays, practical applications draw attention to function effectiveness, performance requirements, and energy efficiency, but of course, there is a trade-off between level of perfection and energy efficiency. Despite the fact that multi-agent patrolling has been investigated from various perspectives over the years, most of the studies lay emphasis on enhancing the performance in task completeness, and the issue of energy minimization has not been sufficiently studied. In this paper, we intended to solve the multi-agent patrolling problem from the aspect of energy consumption. Taking the area cleaning task as an example, the agents are expected to satisfy the given requirement of cleanliness with the lowest possible energy cost and are not asked to keep the environment as clean as possible.

We present a series of decision-making and behavioral strategies deployed by agents. First, we propose the algorithms for estimating the status of the work environment and evaluating an agent's self-importance, which enable agents to individually judge whether the given requirement of quality is satisfied and to understand their contribution degree regarding the system's purpose based on their recent and expected performance. On top of that, we then introduce two types of energy-saving behaviors, *homing* and *pausing*, for agents to take instead of moving on to the next target. Agents would only perform these behaviors when they consider that the given requirement level of task completeness has been reached. In addition, the more important they consider they are for the system, the higher the probability is that the agents perform these behaviors. Besides, we also extended our previous learning method so that agents are able to autonomously select appropriate target decision strategies during planning by monitoring the local energy consumption and number of handled events.

The outline of the following chapters is as follows. Section 2 discusses some related work and our previous work. Section 3 addresses the purpose of our study. Section 4 describes the models of the environment and the agents, along with strategies for selecting targets and generating paths, and then explains the definition of performance measures to clarify the main purpose of our work. Section 5 introduces our proposals for a series of energy-efficient strategies, including a variation of the previous reinforcement learning method. Section 6 shows the experiments conducted to evaluate the proposed methods. The results indicate that our methods enabled agents to individually select appropriate target decision strategies and, more importantly, to reduce the energy cost while cooperatively maintaining the required levels of perfection. Finally, Section 7 summarizes the results and suggests some topics as further research.

2 Related Work

A number of studies have been devoted to multi-agent reinforcement learning, and the research in the multi-agent patrolling field have been exponentially growing over the last decade. Ahmadi and Stone (Ahmadi and Stone, 2005) defined the formulation of a continuous area sweeping task and introduced an initial approach that non-uniformly visits the environment to minimize the estimated cost. They then extended the approach to a multi-robot scenario, where area partitioning by negotiation among agents was conducted (Ahmadi and Stone, 2006). Moreira et al. (Moreira et al., 2009) argued that multi-agent patrolling can be a good benchmark for multi-agent systems and proposed a software simulator constructed strictly for the patrolling tasks. Santana et al. (Santana et al., 2004) solved the multi-agent patrolling problem using reinforcement learning by automatically adapting the strategies of agents to the environment. Portugal and Rocha (Portugal and Rocha, 2013) proposed a distributed approach based on Bayesian interpretation, which effectively solved the multi-robot patrolling problem with scalability and fault-tolerance. Acevedo et al. (Acevedo et al., 2013) described a distributed approach for patrolling missions in irregular-shaped areas with heterogeneous aerial vehicles.

Due to the complexity of actual situations, various studies on reasoning coordination and cooperation between multiple robots were conducted. Hennes et al. (Hennes et al., 2012) provided a collision avoidance system for multiple robots based on a velocity obstacle paradigm. Dinnissen et al. (Dinnissen et al., 2012) developed an algorithm capable of deciding when and how the maps should be merged for solving the multi-robot simultaneous localization and mapping problem using reinforcement learning. Korsah et al. (Korsah et al., 2013) proposed a taxonomy that handles the issues of interrelated utilities and constraints for task allocation problems. Liu and Shell (Liu, 2012) also introduced a dynamic approach for realizing large-scale partitioning.

Concerning the multi-agent CCPP, Yoneda et al. (Yoneda et al., 2015) proposed the *adaptive meta-target decision strategy* (AMTDS), which is the autonomous reinforcement learning of target decision strategies for coordination. With this method, agents investigate different strategies and individually identify the most effective ones to achieve perfect quality. Then, they improved the method by introducing self-monitoring to avoid performance degradation due to over-selection. Sugiyama et al. extended the method by incorporating environmental learning for agents to perform patrolling tasks without knowledge about the important regions (Sugiyama and Sugawara, 2015), simple negotiation for task allocations for prompting division of labor (Sugiyama et al., 2016), and learning of appropriate activity cycle (Sug,).

However, energy usage was not taken into consideration in the studies mentioned above, so agents made an all-out effort and concentrated on performing the tasks perfectly by disregarding energy efficiency.

As regards the issue of energy conservation, only a few studies about multi-agent systems have handled it. Mei et al. (Yongguo Mei et al., 2006) presented an energy-efficient motion planning approach for robot exploration, which selects the next target node based on orientation information and reduces repeated coverage. Cabreira et al. (Milech Cabreira et al., 2018) proposed an energy-aware decentralized real-time search approach for cooperative patrolling problem using multiple unmanned aerial vehicles. The method saves energy by minimizing the number of turns and replaces the centralized decision process using internal matrices and synchronization schemes to merge individual information. In contrast, this paper discusses on energy-aware strategies from the viewpoint of action plans for CCPPs subject to requirement of task completeness.

3 Purpose of the research

The contribution of this paper is to propose a series of decision-making and behavioral strategies for agents to improve energy efficiency. With regard to real-world applications, in comparison with accomplishing the tasks perfectly by ignoring energy usage, people usually place a higher value on reduction of energy cost. For instance, in area cleaning tasks, it is not necessary to maintain the environment extremely clean all the time. Instead, we would prefer the agents to cooperatively satisfy the given requirement of cleanliness with the lowest possible energy cost. Different from previous work, our proposals solve the energy-aware multi-agent patrolling problem by letting agents estimate the status of the environment and judge their own contribution degree to avoid unnecessary movement. We also extended the previous reinforcement learning method to enable agents to learn appropriate target decision strategy individually, keeping in mind the energy efficiency. In addition, several extra experiments were conducted to examine the tolerance degree of our algorithms when the value of quality requirement changes.

4 Model Description

We use the multi-agent CCPP model (Sugiyama et al., 2016), in which multiple autonomous agents move around the work environment and visit locations with required and non-uniform frequencies for given purposes. Depending on the tasks, agents require different capabilities and functionalities, such as maintaining cleanliness in area cleaning tasks and keep safety in security patrolling applications. There are several important assumptions in our work:

- Agents know the structure of the environment.
- Agents have the information about their own position and others' positions.
- More than one agents being at the same position(node) is allowed.
- Agents cannot acquire all the information and knowledge about others.

An environment with the first assumption can be realized by applying algorithms for map creation (Hahnel et al., 2003; Wolf, 2005) and map merging (Dinnissen et al., 2012). The second one is hold when considering a system of agents equipped with indicators, such as infrared emission and reflecting devices, and a receiver which identified their locations and periodically broadcast these data to all agents. Next, allowing multiple agents being at the same node is impossible in the real-world applications, while many noticeable collision avoidance methods (Hennes et al., 2012; Bruni et al., 2013; Chen et al., 2017) have been proposed. Finally, as previously mentioned, there is an important constraint on the multi-agent system that agents may not at any given time know everything about the world of other agents. Therefore, sophisticated coordination should be avoided since agents may have restricted resources including limited CPU power and battery capacity. It is reasonable to suppose that agents must independently creates its action plans based on local view and shallow coordination, by which the agent can only exchange superficial data but does not acquire deep knowledge including other agents' plans, long-term targets, and learned knowledge.

4.1 Environment

Agents move and work in an environment described by graph $G = (V, E)$, where $V = \{v_1, ...v_m\}$ is the set of nodes with coordinates $v = (x_v, y_v)$, and E is the set of edges which agents traverse. It is reasonable to set the length of edges in E to one by adding dummy nodes if necessary. The length of the shortest path between two nodes is denoted by $\{d(v_i, v_j) \mid v_i, v_j \in V\}$. We introduce a discrete time unit called *tick*. In one tick, events occur on nodes, agents individually decide their action plan, and they can move to one of the neighboring nodes along the edges then work on the nodes they visit.

We probabilistically describe the ease of the occurrence of events. Each node owns a value of *probability of event occurrence* (PEO) denoted as $\{P_v \mid v \in V, 0 \leq P_v \leq 1\}$. Depending on the purpose of the system, an event could have different definition. In the case of area cleaning tasks, an event corresponds to the accumulation of dirt, so P_v represents the probability that one piece of dirt has accumulated at v per tick. In the case of security patrolling tasks, an event corresponds to the appearance of enemies or suspicious events, so P_v indicates the probability with which something dangerous has happened and the security level has increased at v per tick. The number of unhandled events on v at time t expressed as $L_t(v)$ is updated based on P_v every tick by

$$L_t(v) \leftarrow \begin{cases} 0 & \text{if an agent has visited } v \text{ at } t, \\ L_{t-1}(v)+1 & \text{if an event occurs with probability } P_v \text{ at } t, \\ L_{t-1}(v) & \text{otherwise.} \end{cases} \tag{1}$$

If an agent has visited v at t, the agent processes the events on v, so $L_t(v) = 0$. A higher P_v means that events easily occur on the node, and that the node is more important. Environments with different characteristics can be expressed using these probabilities.

4.2 Agent

Let $A = \{1, ..., n\}$ be a set of agents, and $v^i(t) \in V$ be the position of agent $i \in A$ at time t. In this paper, agents are given the values of PEO, P_v, in advance but do not know the actual value of $L_t(v)$. Instead, they estimate it by calculating the expected value, $EL_t(v)$, from P_v and $t_{visit}(v)$, the most recent time any agent (may not be i) visited and worked on the node v. Agents know $t_{visit}(v)$ since they can obtain the information of others' positions following the above assumption. $EL_t(v)$ at any future time t is defined by

$$EL_t(v) = P_v \cdot (t - t_{visit}(v)). \tag{2}$$

Note that even if agents are not given P_v in advance, they can learn through experience during patrolling (Sugiyama and Sugawara, 2015).

Agents have their own rechargeable batteries and one or more charging bases. The battery in agent i is denoted by tuple $(B^i_{max}, B^i_{cons}, k^i_{charge})$, where $B^i_{max} > 0$ is the maximal capacity of the battery, $B^i_{cons} > 0$ is the consumption rate per tick when i is moving, and $k^i_{charge} > 0$ is the constant indicating the speed of charge. Let $b^i(t)$ represents the remaining battery capacity in i at time t. When i moves, $b^i(t)$ is updated every tick by

$$b^i(t+1) \leftarrow b^i(t) - B^i_{cons}. \tag{3}$$

When i charges its battery at the charging base, v^i_{base}, the required time for a full charge starting from t is proportional to the amount of battery consumption:

$$T^i_{charge}(t) = (B^i_{max} - b^i(t))/k^i_{charge}. \tag{4}$$

We assume that agents consume B^i_{cons} of battery every time they move, regardless of the number of handled events. Therefore, the amount of energy consumption by agent i from time $t-1$ to time t is defined by

$$E_t(i) = \begin{cases} 0 & \text{if } i \text{ is charging or stays at the same place at } t, \\ B^i_{cons} & \text{otherwise.} \end{cases} \tag{5}$$

The parameters B^i_{max}, B^i_{cons}, and k^i_{charge} can be independent of i, but they are set to same values in this paper as we assume homogeneous agents for simplicity.

According to the above definitions, periodical return to charging bases is required for agents to ensure continuous patrolling, meaning that they must return to v^i_{base} before $b^i(t)$ becomes zero. Since agents know G, they can calculate the potential for each node, which indicates the minimal amount of battery required to return to the charging bases, in advance. Agent i calculates the potential, $\mathscr{P}(v)$, for node v by

$$\mathscr{P}(v) = d(v, v^i_{base}) \cdot B^i_{cons}. \tag{6}$$

A node v is considered *safe* for i at time t if the following condition is satisfied:

$$b^i(t) \geq \mathscr{P}(v) + d(v^i(t), v) \cdot B^i_{cons}. \tag{7}$$

If v does not satisfy the condition, v is considered *unsafe*. When agents create their action plans, they have to avoid running-out of battery by checking whether the next node they are moving to is safe.

4.3 Planning in Agents

Agents create the plans for their paths in two stages: *target decision* and *path generation*. The agent decides the target node in the former stage and generates the appropriate path from the current node to the target node. There are lots of algorithms to determine targets and paths. We use several simple strategies as proposing planning algorithms was not part of our main purpose.

4.3.1 Target Decision Strategies

In the former stage, agent i decides the target node, v_{tar}^{i}, based on (1) on which node the largest number of events is expected to occur or (2) which node in unlikely to be visited by other agents in a short amount of time.

With single strategy regime, agents adopt one of the following strategies to select the target node. By contrast, each agent with the proposed meta-strategy, which is described in the next chapter, independently learns to identify the appropriate strategy from the four strategies through reinforcement learning. We use several simple strategies as proposing planning algorithms was not part of out main purpose.

Random Selection (R): Agent i randomly selects v_{tar}^{i} among all nodes V.

Probabilistic Greedy Selection (PGS): Agent i estimates the value of expected number of unprocessed events and select the one with the highest value. Let $V_g^t \subset V$ be the set of $N_g > 0$ nodes with the maximal values of $EL_t(v)$ for time t. i randomly selects v_{tar}^{i} from V_g^t, where randomness is introduced to avoid a high concentration of the targets selected by multiple agents.

Prioritizing Unvisited Interval (PI): Agent i selects the node that have not been visited recently. i randomly selects v_{tar}^{i} from the set $V_p^t \subset V$, which includes $N_i > 0$ nodes with the highest values of the time difference between current time t and $t_{visit}(v)$. In the similar manner as PGS, randomness is introduced to avoid a high concentration of the targets.

Balanced Neighbor-Preferential Selection (BNPS): BNPS is an advanced version of PGS. The idea is that if agent i estimates that there exist nodes with higher values of expected unprocessed events in the neighborhood using the learned threshold, i selects v_{tar}^{i} from those nodes. Otherwise, i selects v_{tar}^{i} using PGS. A detail explanation is described in Yoneda et al.'s study (Yoneda et al., 2015).

Before agent i generates the path to v_{tar}^{i}, it checks the amount of remaining battery to makes sure that v_{tar}^{i} is reachable. Otherwise, i sets v_{tar}^{i} to its charging base, v_{base}^{i}, and returns to charge its battery. We use *gradual path generation* (GPG) method as path generation strategy. Agent i first calculates the shortest path from current node to v_{tar}^{i} and then, if it estimates that there exist nodes with larger number of unhandled events near the path, it regenerates the path to visit them. The method was chosen because the research by Yoneda et al. (Yoneda et al., 2015) has shown that GPG always outperforms the simple shortest path strategy.

4.3.2 Path Generation Strategies

Before agent i generates the path to v^i_{tar}, it checks $b^i(t)$ and $\mathscr{P}(v^i_{tar})$ in advance to confirm whether v^i_{tar} is reachable. Otherwise, i changes v^i_{tar} to v^i_{base} and then generates a path to return and charge its battery.

Agents use the *gradual path generation* (GPG) method as path generation strategy. In general, agents move along the shortest path, but if there are nodes with larger number of unprocessed events near the path, agents take a detour going to these nodes and deal with these unprocessed events.

Suppose that agent i has its v^i_{tar} determined and sets the 0-th sub-target node as $\mathbf{v}_0 = v^i_t$. Then i recursively selects the node which has the highest value of $EL_t(v)$ from the $(n-1)$-th sub-target nodes as the n-th sub-target node, $\mathbf{v}_n$. Given $d_{cl} > 0$ defined as closeness to select the next sub-target and parameters $0 < k_{att}] \leq 1$ and $1 \leq k_{rover}] < 2$ defined as how long the agent wanders off the shortest paths, the node should satisfies the following conditions:

$$d(\mathbf{v}^i_{n-1}, \mathbf{v}^i_n) \leq d_{cl}, \tag{8}$$

$$d(\mathbf{v}^i_n, v^i_{tar}) < k_{att} \cdot d(\mathbf{v}^i_{n-1}, v^i_{tar}), \tag{9}$$

$$d(\mathbf{v}^i_{n-1}, \mathbf{v}^i_n) + d(\mathbf{v}^i_n, v^i_{tar}) \leq k_{rover} \cdot d(\mathbf{v}^i_n, v^i_{tar}), \text{ and} \tag{10}$$

$$\mathscr{P}(v^i_{tar}) + B^i_{cons} \cdot (d(\mathbf{v}^i_{n-1}, \mathbf{v}^i_n) + d(\mathbf{v}^i_n, v^i_{tar})) \leq b^i(t), \tag{11}$$

However, the process terminates if $d(\mathbf{v}^i_{n-1}, v^i_{tar}) \leq d_{cl}$ or $\mathbf{v}^i_n = v^i_{tar}$. Eventually, the resulting path is generated by connecting $\mathbf{v}^i_{n-1}\mathbf{v}^i_n$ with the shortest path.

Note that agents are only allowed to deploy shallow communication so that they do not have information about the plans, including paths and targets, of others. Therefore, the target of an agent may be cleaned by other agents before it arrives there.

4.4 Performance Measures

Our purpose is to minimize the overall energy cost on the premise of satisfying the requirement for task completeness, which corresponds to the total amount of unprocessed events in the work environment. Accordingly, we evaluate the proposed methods in two aspects: *level of task completeness*, D_{t_s,t_e}, and *total energy consumption of agents*, C_{t_s,t_e}, for a certain time interval (from t_s to t_e).

The level of completeness can have distinct definitions depending on the purposes of the application. In the case of area cleaning tasks, agents are requested to reduce the amount of dirt remaining in the environment and clean up the dirt as soon as possible, so D_{t_s,t_e} is expressed as the cumulative existence duration of dirt. Accordingly, $D_{t_s,t_e}(s)$ is defined by

$$D_{t_s,t_e}(s) = \sum_{v \in V} \sum_{t=t_s+1}^{t_e} L_t(v), \tag{12}$$

where s is the strategy adopted by agents. In the case of security patrolling applications, agents are asked to keep the maximum security level as low as possible, so D_{t_s,t_e} is expressed as the maximal number of unaware dangerous events. Therefore, D_{t_s,t_e} is defined by

$$D_{t_s,t_e}(s) = \max_{v \in V, t_s < t \leq t_e} L_t(v) \tag{13}$$

Besides, C_{t_s,t_e} is defined in the same way for all types of applications by

$$C_{t_s,t_e}(s) = \sum_{i \in A} \sum_{t=t_s+1}^{t_e} E_t(i). \tag{14}$$

Even though smaller values of these measures are considered better, there is still a trade-off between level of perfection and energy cost. In our energy-aware CCPP model which subjects to quality requirements, agents are expected to cooperatively conduct the tasks to the requested extent with less energy. Given a value of *requirement level*, $D_{req}^{|A|} > 0$, instead of minimizing $D_{t_s,t_e}(s)$, agents work towards to minimize $C_{t_s,t_e}(s)$ and keep $D_{t_s,t_e}(s)$ small enough to satisfy the condition $D_{t_s,t_e}(s) \leq D_{req}^{|A|}$ at the same time.

5 Methodologies

Our proposal includes a succession of decision-making algorithms which are called while agents execute their actions according to the generated plans. First, we present the algorithms for estimating whether the given requirement of quality is reached and evaluating agent' self-importance. Based on the preceding results, agents decide the following action by taking into account the status of the environment and themselves. Next, we propose two behavioral strategies adopted by agents as a substitute for moving to the next target with the intention of reducing the energy cost. Finally, we present a variation of the previous method AMTDS (Yoneda et al., 2015). Following is a list of our algorithms:

- Requirement Estimation (with Local Observations)
- Self-Importance Evaluation
- Energy-Saving Behaviors
- Autonomous Meta-Target Decision Strategy for Energy Saving and Cleanliness

Fig. 1 shows the flowchart of the action selection process in agents with the proposed methods. As shown in the flowchart, several conditions should be satisfied for the agents to perform the energy-saving behaviors.

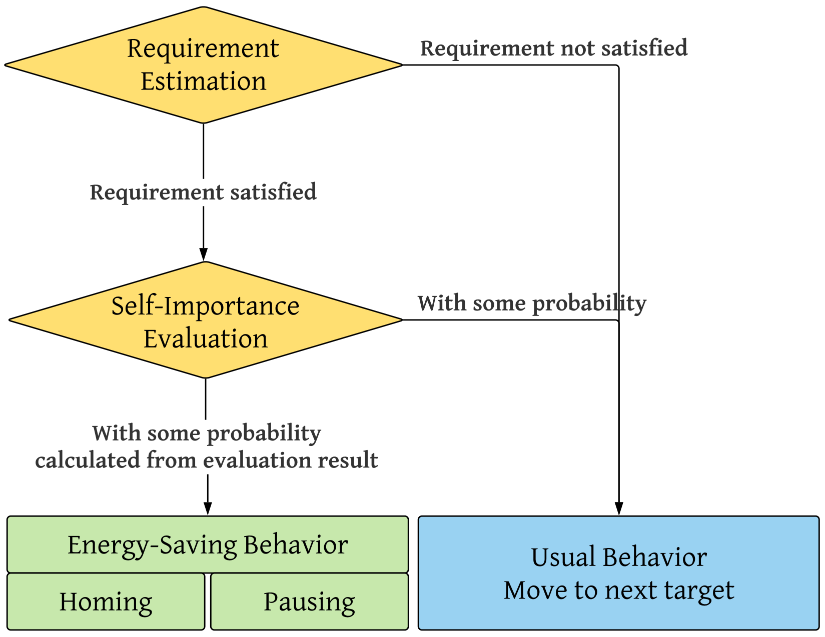

Requirement Estimation
Requirement not satisfied
Requirement satisfied
Self-Importance Evaluation
With some probability
With some probability calculated from evaluation result
Energy-Saving Behavior
Homing
Pausing
Usual Behavior Move to next target

Figure 1: Action selection in agents.

5.1 Requirement Estimation

As we expect the agents to reduce energy cost while satisfy the given requirement at the same time, it is necessary for agents to estimate the current status of the environment to decide the next action. Following our model assumptions, each agent independently estimated the total number of unprocessed events and then judges whether the requirement is satisfied on the basis of $D_{req}^{|A|}$. We propose two versions of algorithms for estimation which differs from each other in the way of choosing the reference nodes.

First, we explain the naive version of the algorithms. For agent i at time t and in environment e, i randomly generates $V_{rand}(v^i(t))$, which is the set of N_{range} nodes it has visited, where N_{range} is a positive integer which indicates the number of reference nodes. The estimated value, EV_t^i, is obtained the average value of $EL_t(v)$ in $V_{rand}(v^i(t))$:

$$EV_t^i = \frac{\sum_{v \in V_{rand}(v^i(t))} EL_t(v)}{N_{range}}. \tag{15}$$

With this value, i judges that the requirement has been achieved only when the following condition is satisfied:

$$EV_t^i \leq D_{req}^{|A|}. \tag{16}$$

If so, i then proceeds to *self-importance evaluation*. Otherwise, i selects the next target node with one of the target decision strategies and generates a path to the destination.

5.1.1 Local Observations

The second version was introduced to improve patrolling fairness based on agents' local estimation. We call the first version *requirement estimation* (RE), and named the new algorithm *requirement estimation with local observations* (RE/LO).

For agent i at time t, i generates a set $V_{est}(v^i(t)) \subset V$ comprising N_{range} nodes. Referring to larger number of nodes gives a more accurate estimation result, but also requires more expensive computational resources. The only difference between RE and RE/LO is the range of reference nodes when forming $V_{est}(v^i(t))$. For simple RE, i randomly selects the nodes from the whole work environment. Besides, the agent with RE/LO only selects the nearby nodes when it is far from the charging base. The farther the agent is from the charging base, the smaller range of area is used for estimation. With the minimal length of reference range given as d_{min}, the set of reference nodes is defined by

$$V_{est}(v^i(t)) = \{v \in V \mid d(v, v^i(t) \leq d_{ref})\}, \tag{17}$$

where d_{ref} is the length of reference range calculated by

$$d_{ref} = max(max\{d(v, v^i_{base}) \mid v \in V\} - d(v^i(t), v^i_{base}),\ d_{min}). \tag{18}$$

Then in the same manner as simple RE, the estimated value is obtained from the average of $EL_t(v)$ in $V_{est}(v^i(t))$:

$$EV^i_t = \frac{\sum_{v \in V_{est}(v^i(t))} EL_t(v)}{N_{range}}. \tag{19}$$

Finally, i take further action depending on whether the condition described in equation 16 is satisfied as previously mentioned.

5.2 Self-Importance Evaluation

Before executing the next action, each agent evaluates its self-importance, which also expresses the contribution degree of an agent, to understand how important it will be for the system. An agent determines its importance by taking into account (1) its recent performance and (2) whether it finds the important regions and is possible to process a large number of events in the subsequent behavior.

Agent i evaluates its self-importance, $Imp^i(t)$, by comparing U^i_p (including U^i_s and U^i_l) and U^i_f. U^i_p is its performance in the past from the short- and long-term viewpoints, and U^i_f is its expected performance in the near future. U^i_p and U^i_f are defined as $u^i_{t_s,t_e}$, the actual or expected number of handled events per tick from time t_s to time t_e:

$$U^i_p = u^i_{t_0,t_c} = \frac{\sum_{t_0 < t \le t_c} L_t(v^i(t))}{t_c - t_0},\ t_0 = \begin{cases} t_c - T_s & \text{short term} \\ t_c - T_l & \text{long term} \end{cases} \tag{20}$$

$$U^i_f = u^i_{t_f,t_c} = \frac{\sum_{t_c < t \le t_f} EL_t(v^i(t))}{t_f - t_c}, \tag{21}$$

where t_c is the current time, T_s and T_l $(T_s < T_l)$ are fixed integers, t_f is the future time when i arrives at the next target, $L_t(v^i(t))$ is the number of events processed by i at time t, and $EL_t(v^i(t))$ is the expected number of events i will handle at future time t. Note that to calculate U^i_f, i needs to select the next target in advance. Finally, $Imp^i(t)$ is obtained by

$$Imp^i(t) = \begin{cases} \frac{U^i_s + U^i_f}{U^i_l} & \text{if } U^i_s + U^i_f \le U^i_l, \\ 0 & \text{if } U^i_l = 0, \\ 1 & \text{otherwise.} \end{cases} \tag{22}$$

Obviously, $0 \le Imp^i(t) \le 1$.

5.3 Energy-Saving Behaviors

We propose two behavioral strategies aiming to reduce energy, which are named *homing* and *pausing*, that agents adopt instead of moving to the next node. An agent will have a chance to perform these behaviors only when it supposes that the given requirement is achieved based on local estimation. These strategies target at reducing agents' activity by avoiding unnecessary movement at different timing. Note that in this paper, only one of the energy-saving strategies was applied during one experimental simulation. The probability of performing the energy-saving behavior is calculated from the result of self-importance evaluation by

$$P^i(t) = 1 - Imp^i(t). \tag{23}$$

5.3.1 Homing

This strategy aims to save energy by making agents stop patrolling and return back to the charging base. Every time after agent i continuously travels for $T_{check} > 0$ ticks, it conducts requirement estimation and self-importance evaluation to decide whether to perform the homing behavior. In addition, i checks the remaining capacity of its battery $b^i(t)$ so that the action will only be taken under the constraint $b^i(t) < k_{homing} \cdot B^i_{max}$, where $0 < k_{homing} < 1$. The constraint is added to prevent agents from frequently returning to the charging base before they work enough. By performing the homing behavior, i immediately sets v^i_{tar} to v^i_{base} to go back and charge its battery. On its way back, i keeps working but does not conduct requirement estimation or self-importance evaluation.

5.3.2 Pausing

The purpose of this strategy is to prolong the time an agent stays at its charging base even after it is fully charged. Unlike the homing behavior, requirement estimation for the pausing behavior is conducted every time after agent i has its battery fully charged. If it decides to perform the pausing behavior according to the estimation and evaluation results, i simply stays at v^i_{base} for another $T_p > 0$ ticks without processing any events. There is no limitation on the number of pausing behavior performed consecutively, which means that it is possible for an agent to adopt the pausing behavior over and over again based on the results of local estimation.

5.4 Autonomous Strategy Selection

As our main purpose is the reduction of overall energy cost, we extend the learning method AMTDS (Yoneda et al., 2015) and name the new method *AMTDS for energy saving and clean-*

liness (AMTDS/ESC). With the extended method, agents choose appropriate target decision strategies from the number of handled events per unit of energy using reinforcement learning. A larger number of handled events and a smaller energy cost are preferred.

Suppose that agent i selects v^i_{tar} with strategy s, where s is one of the target decision strategies described in the previous section. After i moves to v^i_{tar} along the path generated by GPG, it calculates the reward of s by

$$r^i_{t_0,t_0+d_{travel}} = \frac{\sum_{t_0<t\le d_{travel}} L_t(i) / \sum_{t_0<t\le d_{travel}} E_t(i)}{d_{travel}}, \tag{24}$$

where d_{travel} is the length of travel from time t_0 when i started until the time it arrived at its target. Subsequently, the Q-value of s is updated as

$$Q(s) \leftarrow (1-\alpha)\cdot Q(s) + \alpha \cdot r^i_{t_0,t_0+d_{travel}}. \tag{25}$$

As described above, to gain higher rewards, agents choose the appropriate strategy with which they can process more events using less energy. In other words, AMTDS/ESC enables agents to learn to select the strategy that minimizes the energy cost and maximizes the number of handled events per tick at once. In addition, the ε-greedy method is used during learning.

6 Experiments and Analysis

The proposed methods are applied to an area cleaning application by multiple autonomous agents and evaluated in a simulation environment. Therefore, agents move around the area and clean the dirt on the nodes they visit, and the given quality requirement indicates the lowest total amount of dirt in the environment. We experimentally demonstrate that the proposals enable agents to cooperatively reduce energy cost and satisfy the given requirement of total dirt amount at the same time.

6.1 Experimental Settings

We prepared a large and complex environment consists of a corridor and six rooms labeled by *Room N* (where N = 0,...5) with different characteristics. The environment is represented by a two-dimensional grid space with several obstacles, where G is defined as a 101×101 grid. The distribution of colored regions is shown in Fig. 2.

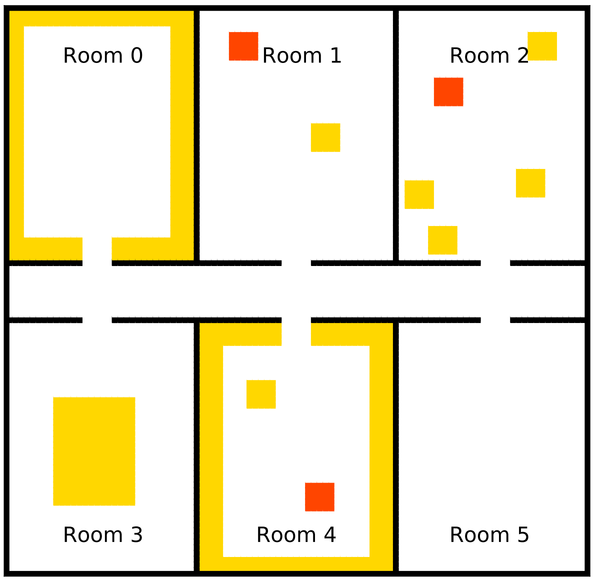


Figure 2: Experimental environment.

Dirt is accumulated uniformly in Room 5, and easily accumulated near the wall in Room 0. In contrast, Rooms 1, 2, and 3 are more complicated environments with square regions that are easily contaminated, and Room 4 has both of these characteristics. Accordingly, we set the probability of dirt accumulation P_v for any node $v \in V$ as

$$P_v = \begin{cases} 0 & \text{if } v \text{ was in a black region,} \\ 10^{-3} & \text{if } v \text{ was in a red region,} \\ 10^{-4} & \text{if } v \text{ was in a yellow region, and} \\ 10^{-6} & \text{otherwise.} \end{cases} \tag{26}$$

We deployed 20 agents in the environment, and all of them are assumed to be homogeneous: they have the same amount of battery capacity, use the same path generation strategy, and adopt one of the five target decision strategies (R, PGS, PI, BNPS, and AMTDS/ESC). When they use AMTDS/ESC, they independently select one strategy from R, PGS, PI, and BNPS based on local learning. The parameter values used in the target decision and path generation strategies are listed in Table. 1. Note that these values are determined by considering the size of the environment and the number of agents but are not optimal.

Table 1: Parameters in target decision and path generation strategies.

Strategies	**Parameters**	**Values**
PGS	N_g	5
PI	N_i	5
BNPS	α	0.1
	d_{th}	15
AMTDS/ESC	α	0.1
	ε	0.05
GPG	d_{cl}	15
	k_{att}	1.0
	k_{rover}	1.2

Agents share the same charging base located in the center of the environment as $v^i_{base} = v_{base} = (0,0)$ for $i \in A$. They start their patrol from v_{base} and must periodically return to recharge their batteries. The battery specifications of all agents are set as $(B^i_{max}, B^i_{cons}, k^i_{charge}) = (2700,3,1)$, meaning that an agent could continuously operate for up to 900 ticks and requires 2700 ticks for a full charge when the battery is totally running out of power. Therefore, when all the agents constantly work to make full used of their batteries, the theoretical maximal value of C_{t_s,t_e} per tick will be 15. In the following experiments, we compare the resulting $C_{t_s,t_e}(s)$ to this value in order to evaluate the proposed methods.

To solve the energy-aware CCPP with the proposed methods, agents estimate the status of environment and evaluate the importance of themselves. According to the results, agents perform either the homing or pausing behavior instead of heading toward the next target under certain circumstances. The decision of action selection is individually made by each agent with their local viewpoints. We compare the values of D_{t_s,t_e} and C_{t_s,t_e} every 100 ticks. The experimental results below are the averages of five independent trails with different random seeds, and the length of each trial is 500,000 ticks.

6.2 Behavioral Energy-Aware Strategies

We evaluate our proposals by comparing the performance of the three agent behavioral regimes. The first experiment is the control experiment, in which agents only take usual actions and ignore energy efficiency. In the second experiment, there are chances for agents to perform the homing behavior instead of exploring the environment so that they stop patrolling and return to the charging base and charge. In the third experiment, agents perform the pausing behavior with some probability after their batteries are fully charged so that they rake a rest instead of leaving the charging base for patrolling. In the following experiments, we use the simple RE for estimation.

The values of parameters for requirement estimation, self-importance evaluation, and energy-saving behaviors used in the experiments are listed in Table 2. After experimenting with several different values, the value of quality requirement is determined under the principle of picking a value lower than that when agents make an all-out effort to work so that they are expected to avoid unnecessary movement by taking rest appropriately for saving energy.

Table 2: Parameters in energy-aware strategies.

Methods	**Parameters**	**Values**
Requirement Estimation	N_{range}	100
	D^{20}_{req}	1200
Self-Importance Evaluation	T_s	20
	T_l	50
Homing Behavior	T_{check}	100
	k_{homing}	$\frac{1}{3}$
Pausing Behavior	T_p	20

Fig. 3 compares the performance measures, which are the total dirt amount in the environment and the overall energy consumption of all agents, for each target decision strategy and agent behavior. The dotted red line represents the given requirement of cleanliness D^{20}_{req}, and the dotted green line represents the theoretical maximal value of energy consumption. Note again that the smaller performance values are better.

The results indicate that the proposal of energy-aware strategies successfully saves energy while agents could still satisfy the given requirement of remaining dirt. With the proposed learning method of target decision strategies, AMTDS/ESC, when given a requirement 2.6 times higher than the performance achieved by agents with normal behaviors, the overall energy cost

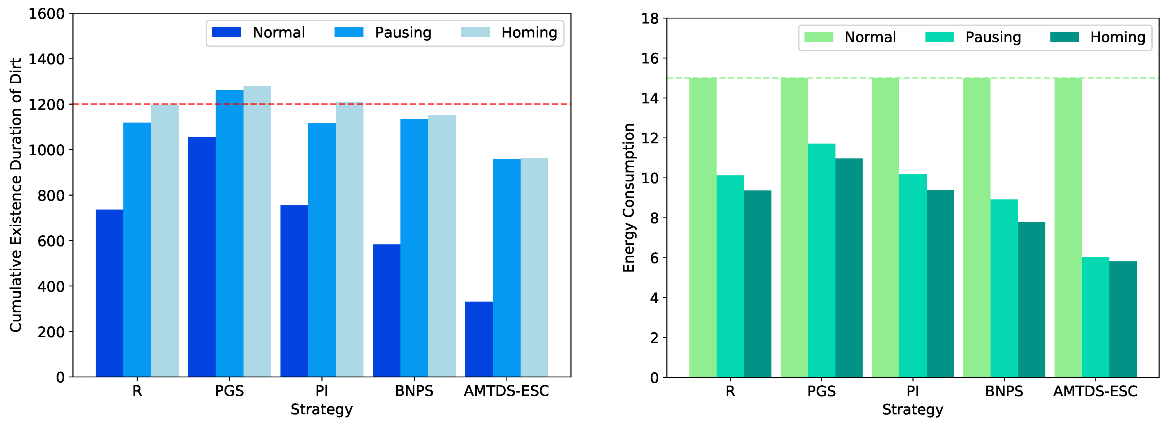


Figure 3: Values of performance measures using RE.

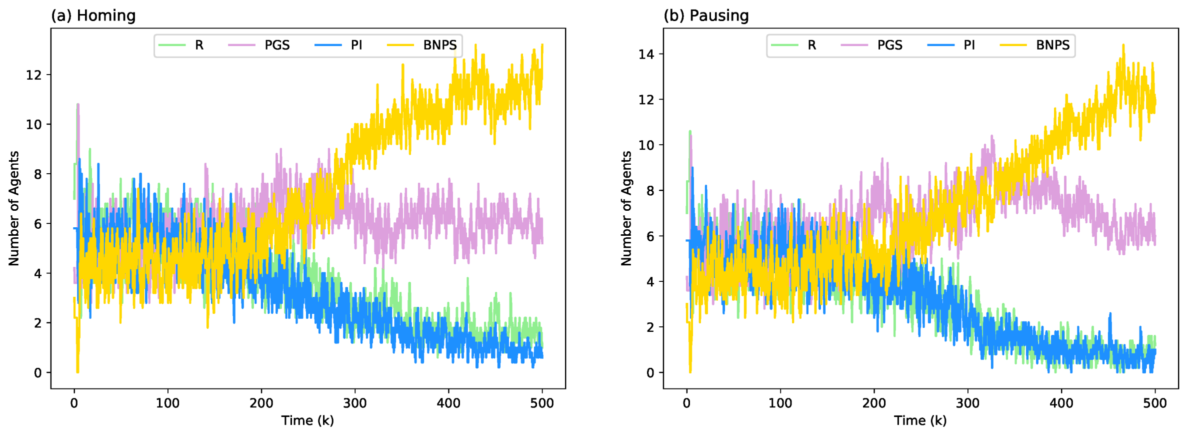


Figure 4: Number of agents with each target decision strategy.

is reduced about 60% with the proposed behavioral strategies. By comparing the two performance metrics, we found that AMTDS/ESC always outperforms other single strategy regimes in respect of lower energy cost. As expected, agents with the meta-strategy select the appropriate target decision strategies locally and respectively, which allows them to cooperatively clean the environment by adopting strategies with different characteristics.

Fig. 4 shows the number of agents with each strategy, which is the learning result of AMTDS/ESC, in both cases of the homing and pausing behaviors. BNPS and PGS were eventually likely to be selected as the target decision strategy by most of the agents. This arises from that with the greedy algorithms, agents can clean the complex environment more efficiently. Also, there are no significant differences between the strategy regime structures of the two energy-saving behaviors.

The homing behavior appears to be more effective than the pausing behavior in terms of lower values of energy consumption, and the values of remaining dirt amount are closer to the given requirement level. A conceivable reason might be the difference in frequencies of conducting requirement estimation. For the homing behavior, requirement estimation is conducted every 100 ticks, which means that it could be called up to 9 times during an operation cycle. On the other hand, requirement estimation could only be conducted after charging for the pausing behavior. For this reason, agents adopting the homing behavior have a higher chance of executing energy-aware plan.

6.3 Requirement Estimation with Local Observations

In the following experiments, our purpose is to investigate the advantages and impact of introducing local observations to the algorithms of requirement estimation. We apply the novel algorithm of requirement estimation, RE/LO, to the two energy-saving behaviors respectively and compare the performances.

First, we would like to ensure that agents with the novel algorithm are still able to solve the energy-aware CCPP. Fig. 5 plots the performance measures of each target decision strategy and agent behavior. In the same manner as Fig. 3, the red and green dotted lines represent the requirement of cleanliness and the theoretical value of energy consumption before applying the energy-aware strategies. The results indicate that after introducing local observations to requirement estimation, agents can cooperatively reduce energy consumption while keeping the value of total dirt amount lower than the given requirement level as before.

Second, we compare the results of RE and RE/LO for each energy-saving behavior to see how the local observations affect the performance. Fig. 6 shows the comparison of the two requirement estimation algorithms in terms of the cumulative existence duration of dirt, and Fig. 7 compares them from the aspect of overall energy consumption. In the case of the homing behavior, agents with RE/LO consume less energy than those with RE, resulting in slightly higher values of dirt amount, which is a more preferable result. On the other hand, in the case of the pausing behavior, agents give almost the same performance with the two algorithms.

At last, we investigate the main advantages of introducing local observations by looking into the ratio of remaining dirt amount in each room to the whole environment. Fig. 8 plots the sum of probability of dirt accumulation P_v in each room, which corresponds to the ease of dirt accumulation. Fig. 9 compares the amount of remaining dirt in each room by each algorithm in percentage. Under the circumstances that we require the agents to conduct the given tasks fairly in terms of patrolling, a more preferred outcome will be having similar values of the ratio of the resulting dirt amount in each room to the probability of dirt accumulation, which means

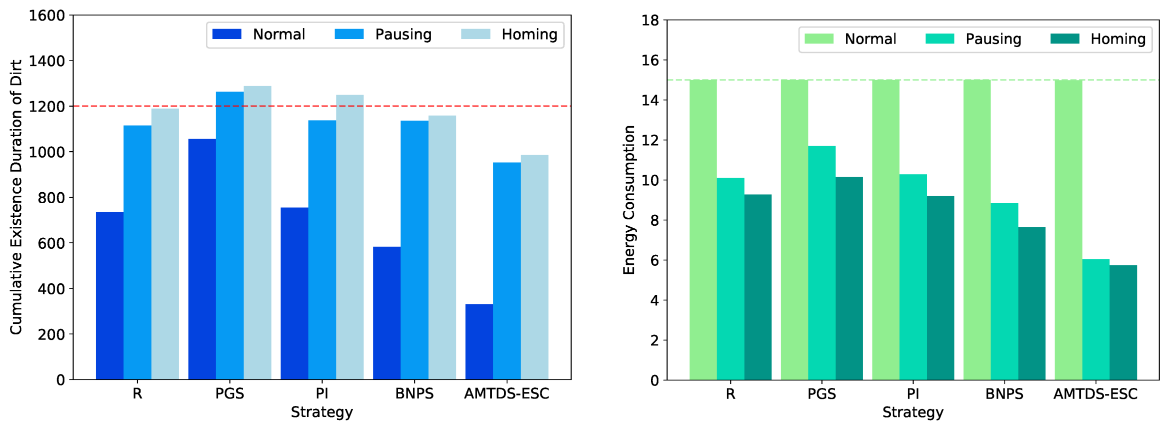


Figure 5: Values of performance measures using RE/LO.

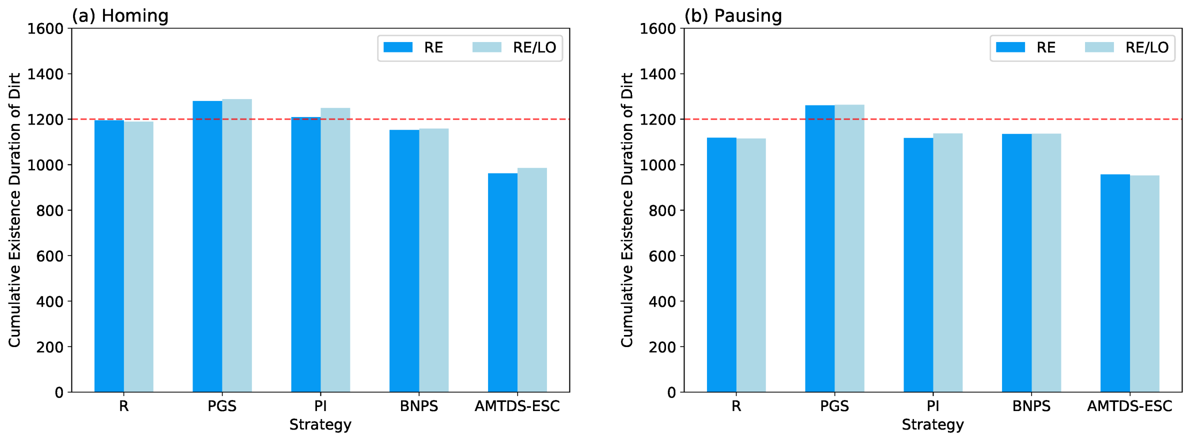


Figure 6: Comparison of RE and RE/LO in D_{t_s,t_e}.

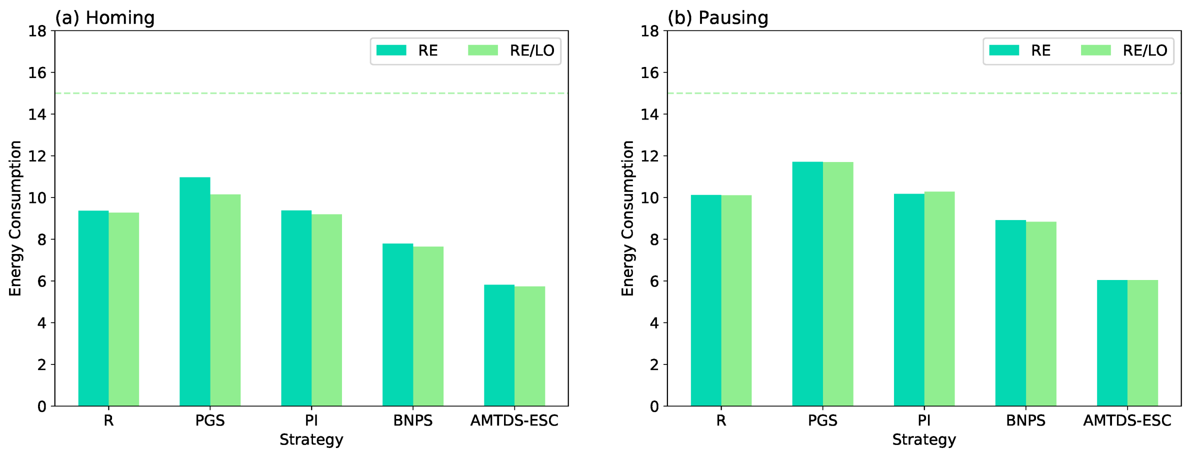


Figure 7: Comparison of RE and RE/LO in C_{t_s,t_e}.

that there is no concentration of agents at the dirty rooms.

Fig. 10 plots the differences of ratio between dirt accumulation probability and the remaining dirt amount in each room when agents deploy AMTDS/ESC as target decision strategy, and Table 3 summarizes the sum of all the rooms for each algorithm and agent behavior. It is quickly shown by those results that RE/LO gives better performance in terms of avoid unfair patrols.

Table 3: Sum of differences between PEO and ratio of remaining dirt amount.

Algorithms	**Homing**	**Pausing**
RE	54.41%	52.23%
RE/LO	41.93%	40.35%

As shown in Fig. 9(a), since agents are given the probability of dirt accumulation in advance and they have the knowledge about dirty regions, those with greedy strategies including PGS and BNPS tend to gather to the dirty rooms such as Room 0 and Room 4 and rarely clean the rooms with low probability of dirt accumulation such as Room 5. As a result, agents patrol in a biased manner and cause the cleanliness of the rooms where dirt hardly accumulates to become worse than those rooms which are supposed to be easily contaminated.

Although introducing local observations does not significantly influence the overall performance as shown in Fig. 6-7, Fig. 9(b) indicates that agents using AMTDS/ESC with RE/LO are able to fairly clean all the rooms so that the resulting amount of remaining dirt in each room is comparatively closer to the sum of accumulation probability than that by agents using AMTDS/ESC with RE. Moreover, agents with RE tend to ignore Room 5, which is not likely to be contaminated and has no dirty regions, while the RE/LO algorithm solves this problem as shown in Fig. 10.

Estimating the total dirt amount from local observations affects decision-making in agents when they are far from the charging base. This somehow prompts agents to work more in the relatively clean rooms instead of going back to the charging base when they judge that the requirement is satisfied from the viewpoint of the whole environment, and thus avoids concentration of agents at the dirty rooms and unfair patrols.

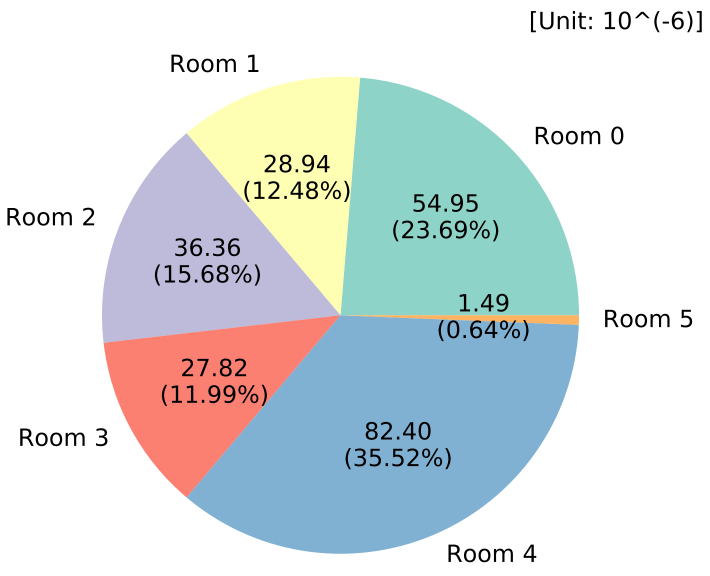


Figure 8: Sum of dirt accumulation probability in each room.

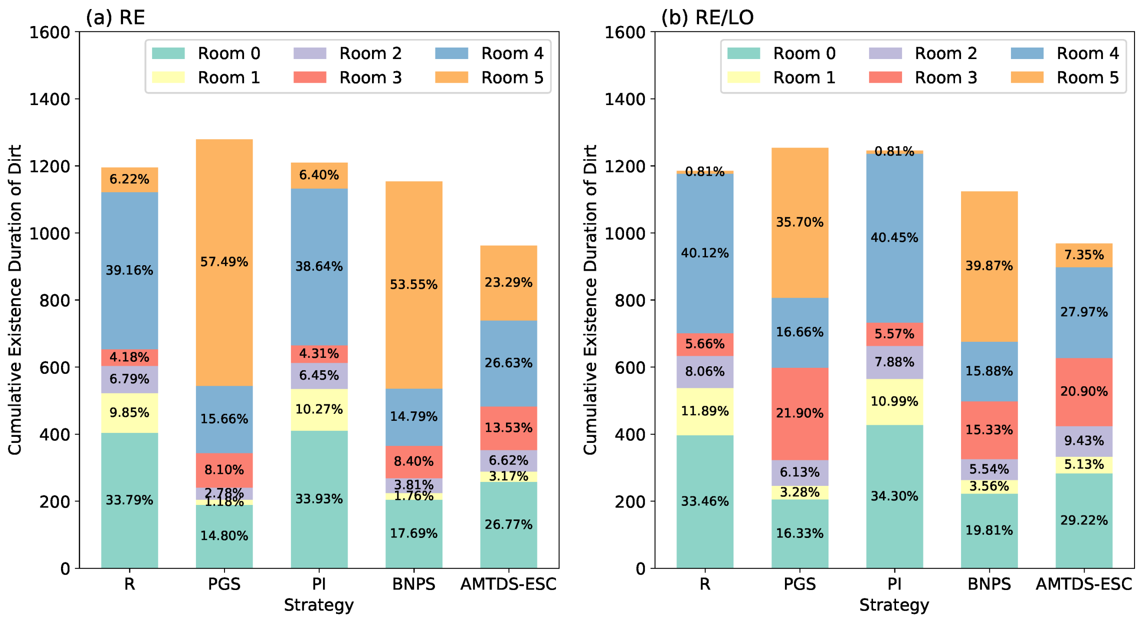
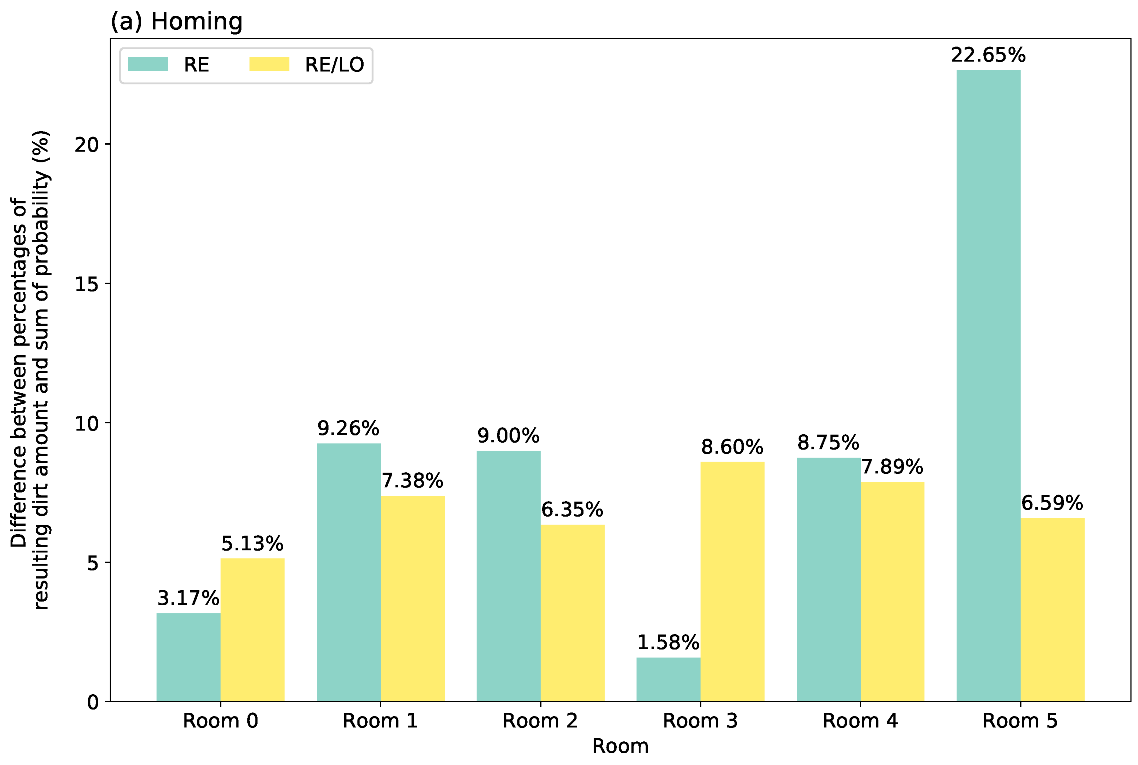


Figure 9: Cumulative existence duration of dirt in each room (Homing).



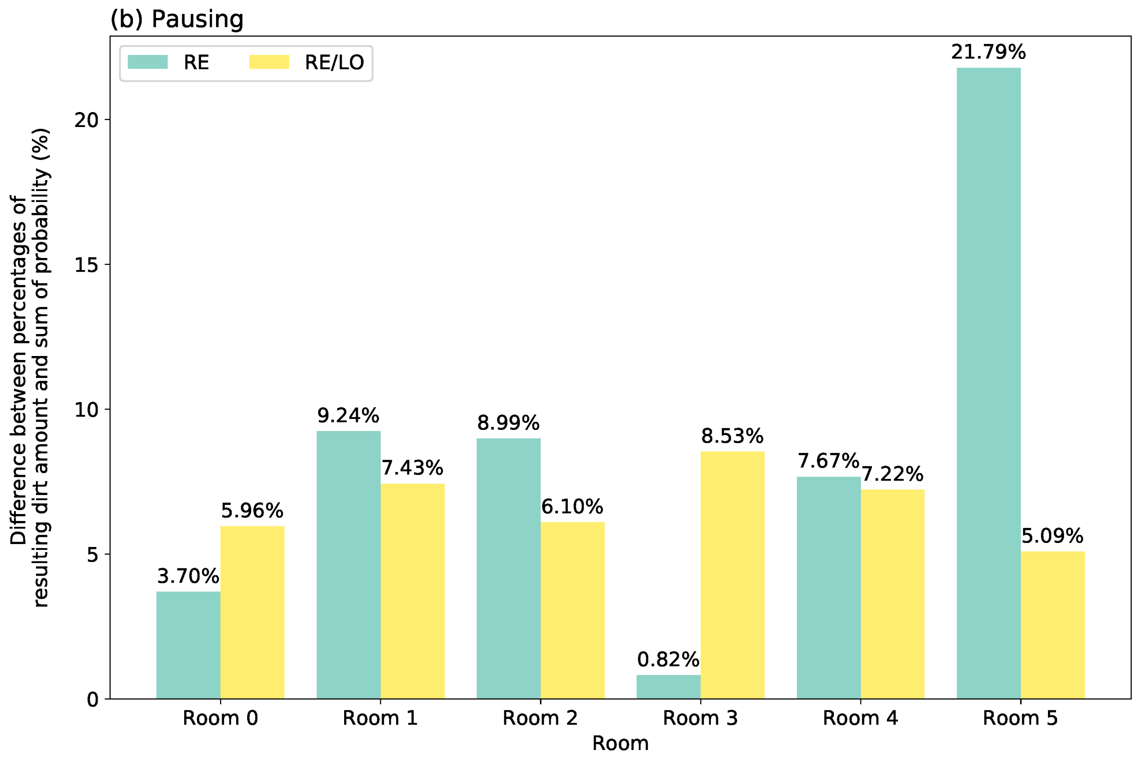


Figure 10: Difference in percentages between PEO and remaining dirt amount in each room.

6.4 Extra Experiment: Visiting of VIP

In real-world applications, the given requirement of quality could change because of various situations. For example, when a visit of celebrities or a very-important-person (VIP) is scheduled, it requires the working environment to be cleaner than usual, which means that the requirement of cleanliness is temporarily increased. Needless to say, there exist opposite circumstances such as during long vacation when the work environment is shut down, the requirement of cleanliness might be loosen to conserve energy. In the following extra experiments, we investigate the degree of tolerance of our proposed algorithms under those kinds of situation.

Two patterns of requirement change are prepared: *800-1200-800* and *1200-800-1200*. The former pattern corresponds to the long vacation case mentioned above, and the latter patter corresponds to the visiting of VIP case. The experiments are conducted for 2,000,000 ticks in total. In the case of the 800-1200-800 pattern, the initial value of quality requirement is set to 800. At 700,000 tick, the value of requirement is changed to 1200 for 500,000 ticks, then is switched back to 800 again at 1,200,000 tick. Likewise, the requirement value is set in the same way for the case of the 1200-800-1200 pattern.

Fig. 11 and 12 plot the results of cumulative existence duration of dirt over time for the two patterns with different energy-saving behaviors respectively. We only discuss the results of AMTDS/ESC among the five target decision strategies as it is part of our proposal. The results indicate that agents with the proposed energy-aware algorithm are capable of handling the change of requirement values. The pausing behavior gives especially good performance that the given quality requirement is satisfied at all times, and the total dirt amount rises and falls as the requirement value changes.

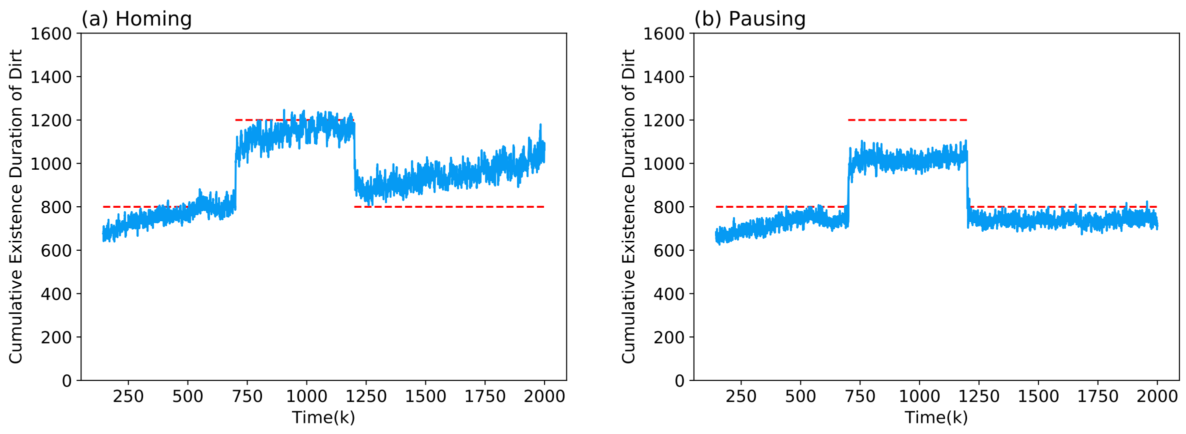
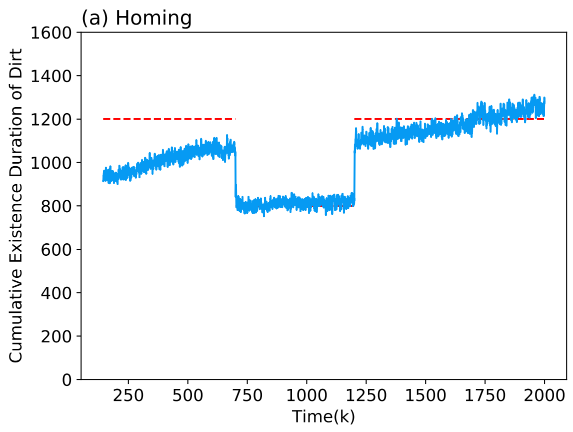


Figure 11: D_{t_s,t_e} over time with the requirement pattern 800-1200-800.



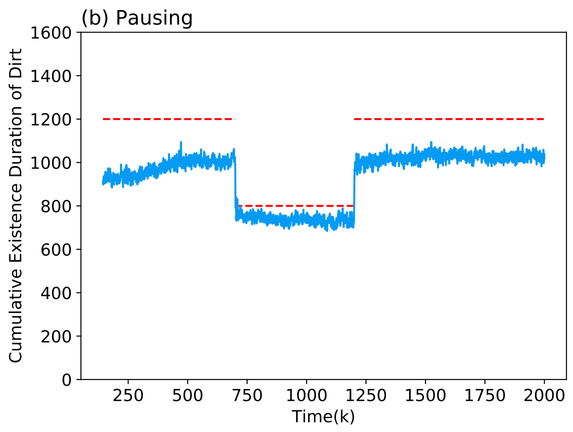


Figure 12: D_{t_s,t_e} over time with the requirement pattern 1200-800-1200.

6.5 Extra Experiment: Algorithms for Updating Positions of Agents

As mentioned in the above chapter of model description, our research assumes that agents can always obtain others' current positions. This knowledge is very important since it is involved in the algorithms of most of the proposed methods including requirement estimation, self-importance evaluation and the target decision strategies. However, discarding this assumption can make the system more practical and meet the definition of decentralized systems better. Thus, instead of providing agents others' positions all the time, we suggest two algorithms for updating positions of agents and examine the proposed energy-aware strategies with those algorithms in the following extra experiments.

We called the usual updating algorithm *full* and named the first proposed algorithm *base*, with which agents can only update others' positions when they are at the charging base. The second algorithm is named *exchange*, with which two agents exchange their position information when they are close enough to each other. The exchangeable distance is set to 20 in the following experiments, where the size of the working environment is 101 $\times$ 101. Note that during the updating process, agents only update the positions of the target agents at that certain time but not the whole visiting history.

Fig. 13 and 14 show the values of total dirt amount and energy consumption for the three position updating algorithms with the two energy-saving behaviors respectively. In terms of the performance measures, agents with the base algorithm give acceptable results. Even though agents with the full algorithm still give the best performance, those with the base algorithm are able to satisfy the given requirement and reduce overall energy cost. Furthermore, in the case of pausing behavior, agents with the base algorithm result in consuming the same amount of energy as those with the full algorithm, while the total dirt amount is far lower than the requirement. This points out that there is still room for improvement and we can expect agents with the base algorithm to give better performance after further research.

In contrast, agents with the exchange algorithm fail to save energy. Fig. 15 compares the values of performance measures when using the exchange algorithm with energy-saving strategies and that when agents only take usual action. The results indicate that agents could not properly adopt energy-saving strategies with the exchange algorithm so that the amount of consumed energy is almost the same as those with normal behaviors, while the remaining dirt amount is higher. Despite the fact that the exchange algorithms follows the definition of decentralized systems more, the algorithm is unsuitable for the proposed energy-saving strategies.

Besides, an interesting result is found when comparing the difference between the dirt accumulation probability distribution and the remaining dirt amount in each rooms. Fig. 16 plots the differences of ratio in percentages between PEO and the remaining dirt amount in each room when agents deploy AMTDS/ESC as target decision strategy, and Table 4 summarizes the sum of all the rooms for each algorithm and agent behavior.

Table 4: Sum of differences between PEO and ratio of remaining dirt amount.

Updating Algorithms	**Homing**	**Pausing**
full with LO	41.93%	40.35%
base	21.16%	22.19%
exchange	28.87%	28.97%

Surprisingly, with the two new algorithms for position updating, agents using RE outperform those using the full algorithm with RE/LO in terms of patrolling fairness. To be more precise, the sum of differences when using the base and exchange algorithms reduce to almost half of that when using the full algorithm. In particular, agents with the base algorithm outperform the rest, and both of the new algorithms give good performance in Room 3, 4, and 5. We conjecture the reason being that with the new updating algorithms, agents rarely have chances to update others' positions. In the case of the base algorithm, when an agent is at the charging base, it can only obtain information of 20 nodes at once, while there are over 10 thousand nodes in total. Owing to this, agents have to create their action plan based on the out-of-date information, so they somehow rely more on the probability distribution, which is given in advance. To conclude, the result obtained is unintentional and it is more reasonable to modify the algorithms so that when agents update the position information, they obtain not only others' current positions, but also the routes that the other agents have visited for a certain period of time.

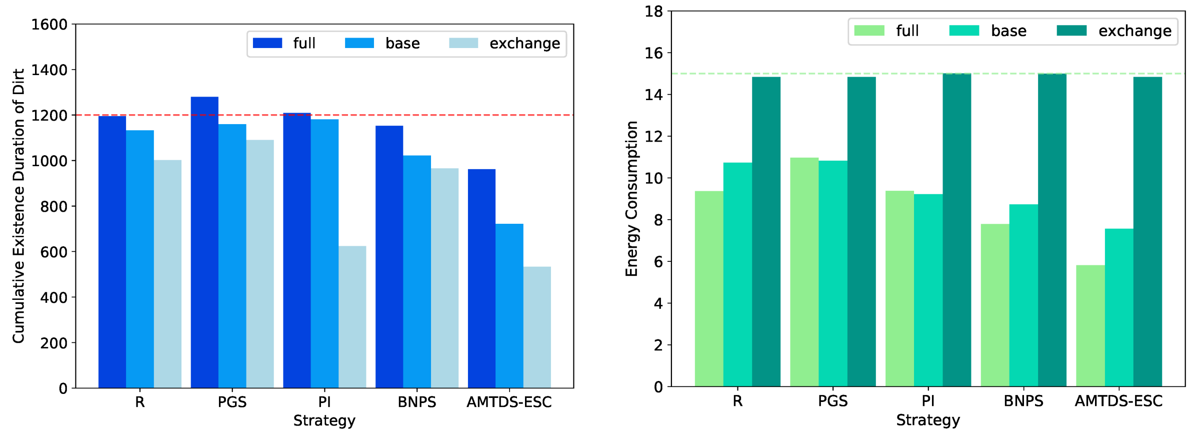


Figure 13: Values of performance measures with the homing behavior.

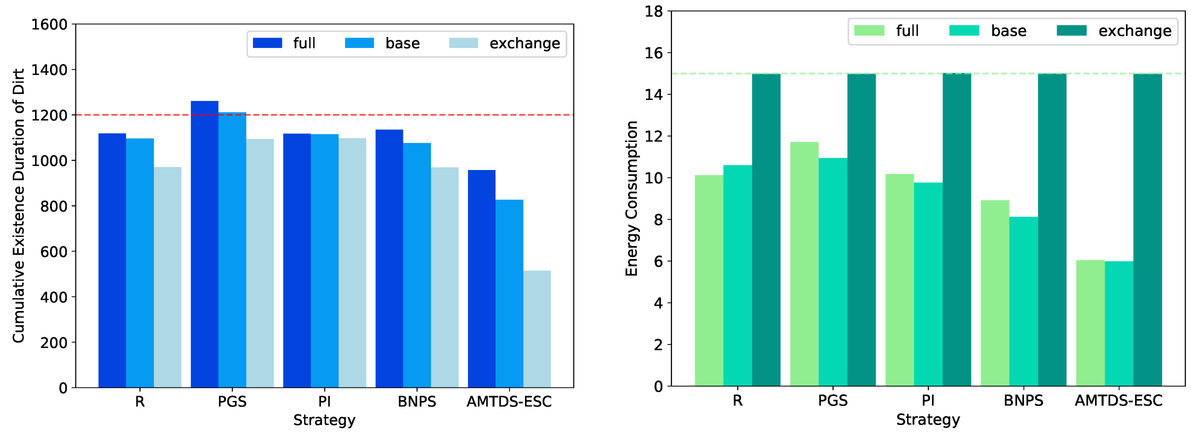


Figure 14: Values of performance measures with the pausing behavior.

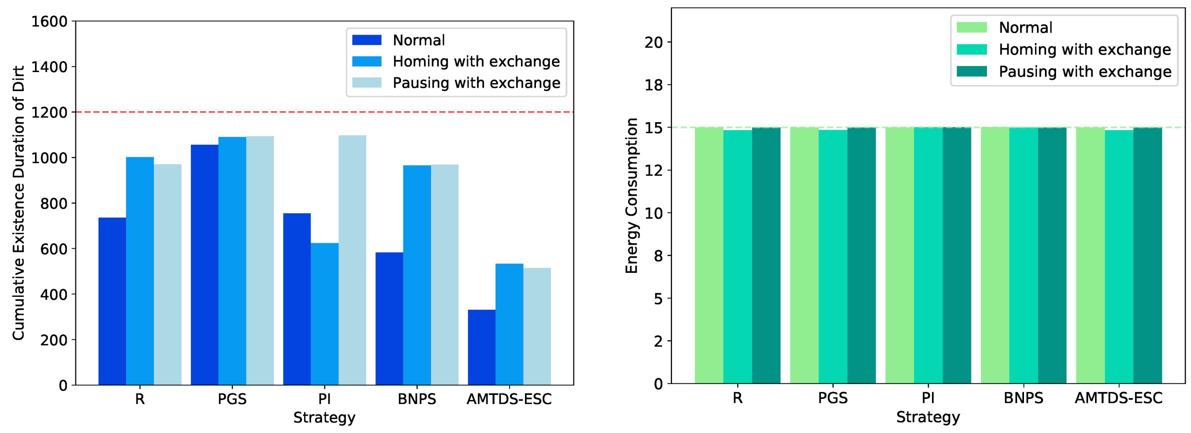
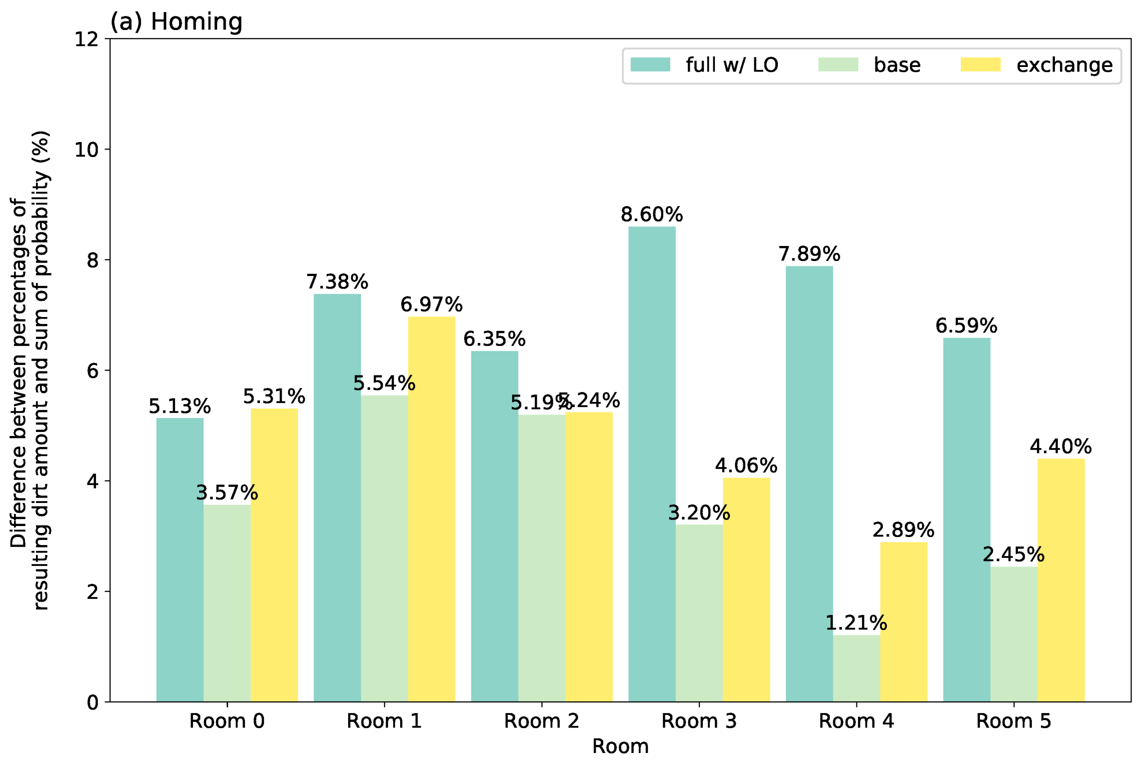


Figure 15: Comparison of performance measures between normal behavior and the exchange algorithm with energy-saving behaviors.



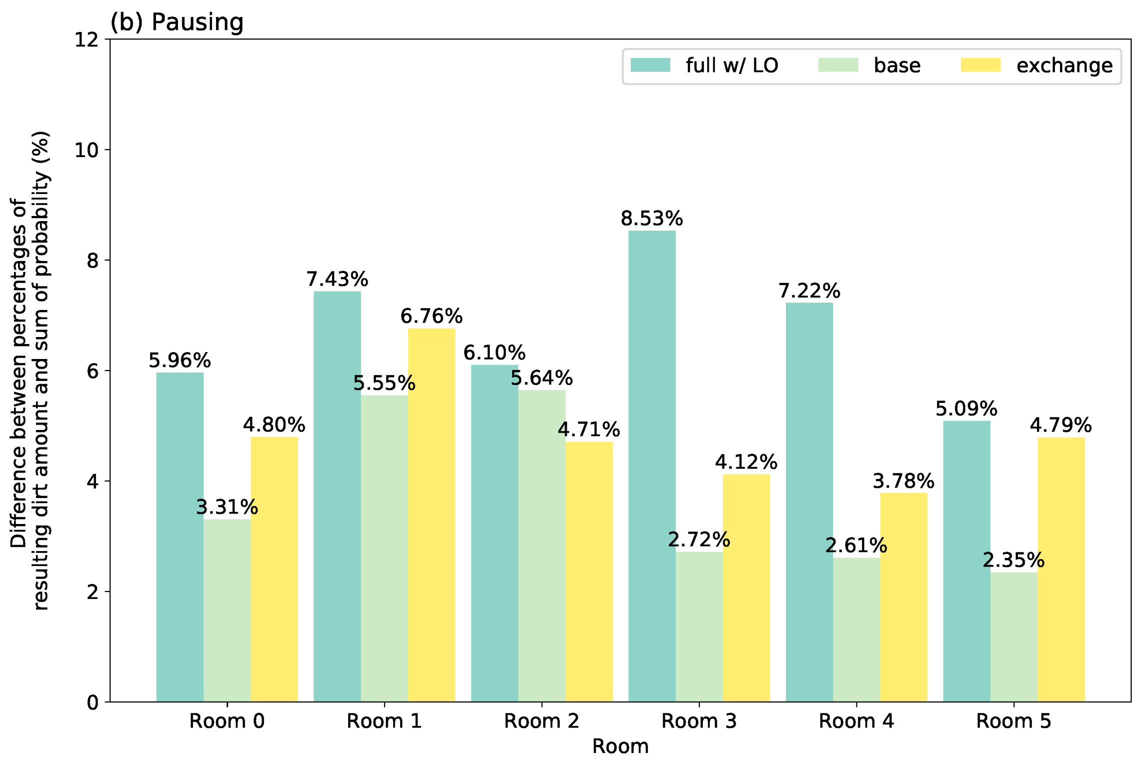


Figure 16: Difference in percentages between PEO and remaining dirt amount in each room.

7 Conclusion and Future Work

7.1 Conclusion

This paper, differently from others, intended to solve the energy-aware multi-agent CCPP subject to quality requirement from the aspect of energy cost reduction with a series of behavioral strategies taken by agents. We extended the learning method of target decision strategies and proposed a sequence of decision-making and behavioral strategies. Agents with the proposed methods independently estimate the current status of the work environment and evaluate their contribution degree with regard to the system. Based on the decision outcome, they then choose to remain performing their usual behavior or to adopt energy-saving plans, including returning to the charging base (homing) or taking a break (pausing). We also introduced an advanced method for requirement estimation to avoid biased patrol and achieve fair task execution.

The experimental results confirmed that the proposed methods enabled agents to reduce the energy cost while cooperatively maintain the given requirement of quality perfection and patrol fairly in a large and complex environment with local observations. Within the five target decision strategies, the proposed method of reinforcement learning method AMTDS/ESC was able to give the best performance in respects of cleanliness and energy consumption. Moreover, concentration of agents at rooms where events are more likely to happen can be avoided by incorporating local observations into requirement estimation. In addition, extra experiments were conducted to examine the tolerance degree of our proposal under the circumstances of changing the given value of quality requirement and to investigate different algorithms for updating positions of agents.

7.2 Future Work

Our future plan covers several different aspects including learning of the environment, combination of the energy-saving behaviors, improvement on agent importance evaluation, and algorithms for updating positions of agents. In this paper, we assume that agents know the structure and event occurrence probability distribution of the environment, and our algorithms of requirement estimation and self-importance evaluation both require the knowledge. Nevertheless, this assumption restricts the range of applicable domain of the proposed method from large scale networks. Hence, more work needs to be done regarding the issue of environment learning so that agents can perform patrolling tasks without knowing PEO in advance. On the other hand, we believe that the combination of homing and pausing behaviors can achieve better performance, therefore we plan to integrate the two strategies as well as learning of parameters such as the length of pausing time and check interval. Moreover, the charging bases for all agents are located in the same place in this paper, whereas letting the charging bases be scattered over the environment or setting up multiple charging bases might make it easier for agents to perform energy-saving behavior. In our research plan, we will also focus on enabling agents to autonomously and individually evaluate their importance with regard to the system from their recent contribution and performance. With this functionality, a continuous system can eliminate old robots and introduce new ones without affecting the overall performance. Lastly, modification of the algorithms for updating positions of agents is also necessary to make the system more practical. Further research should be conducted to analyze how the position updating algorithms influence the results of local observations.

REFERENCES

Acevedo, J. J., Arrue, B. C., Maza, I., and Ollero, A. (2013). Distributed approach for coverage and patrolling missions with a team of heterogeneous aerial robots under communication constraints. *Int. Journal of Advanced Robotic Systems*, 10(1):28.

Ahmadi, M. and Stone, P. (2005). Continuous area sweeping: a task definition and initial approach. In *ICAR '05. Proceedings., 12th Int. Conf. on Advanced Robotics*, pages 316–323.

Ahmadi, M. and Stone, P. (2006). A multi-robot system for continuous area sweeping tasks. In *Proc. 2006 IEEE Int. Conf. on Robotics and Automation (ICRA 2006)*, pages 1724–1729.

Almeida, A., Ramalho, G., Santana, H., Tedesco, P., Menezes, T., Corruble, V., and Chevaleyre, Y. (2004). Recent advances on multi-agent patrolling. In Bazzan, A. L. C. and Labidi, S., editors, *Advances in Artificial Intelligence – SBIA 2004*, pages 474–483, Berlin, Heidelberg. Springer Berlin Heidelberg.

Bruni, L., Colombo, A., and Del Vecchio, D. (2013). Robust multi-agent collision avoidance through scheduling. In *52nd IEEE Conference on Decision and Control*, pages 3944–3950.

Chen, Y. F., Liu, M., Everett, M., and How, J. P. (2017). Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 285–292.

Dinnissen, P., Givigi, S. N., and Schwartz, H. M. (2012). Map merging of multi-robot slam using reinforcement learning. In *2012 IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC)*, pages 53–60.

Hahnel, D., Burgard, W., Fox, D., and Thrun, S. (2003). An efficient fastslam algorithm for generating maps of large-scale cyclic environments from raw laser range measurements. In *Proc. 2003 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS 2003)*, volume 1, pages 206–211.

Hennes, D., Claes, D., Meeussen, W., and Tuyls, K. (2012). Multi-robot collision avoidance with localization uncertainty. In *Proc. of the 11th Int. Conf. on Autonomous Agents and Multiagent Systems - Volume 1*, pages 147–154, Richland, SC. IFAAMAS.

Iocchi, L., Marchetti, L., and Nardi, D. (2011). Multi-robot patrolling with coordinated behaviours in realistic environments. In *2011 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2796–2801.

Korsah, G. A., Stentz, A., and Dias, M. B. (2013). A comprehensive taxonomy for multi-robot task allocation. *The International Journal of Robotics Research*, 32(12):1495–1512.

Liu, L., S. D. (2012). Large-scale multi-robot task allocation via dynamic partitioning and distribution. In *Auton Robot*, volume 33, page 291–307.

Milech Cabreira, T., Stift Kappel, K., Ferreira, P. R., and Brisolara de Brisolara, L. (2018). An energy-aware real-time search approach for cooperative patrolling missions with multi-uavs. In *2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE)*, pages 254–259.

Moreira, D. M. D., Ramalho, G., and Tedesco, P. C. A. R. (2009). Simpatrol - towards the establishment of multi-agent patrolling as a benchmark for multi-agent systems. In *ICAART*.

Panait, L., L. S. C. (2005). Multi-agent learning: The state of the art. In *Autonomous Agents and Multi-Agent Systems*, volume 11, page 387–434.

Portugal, D. and Rocha, R. P. (2013). Distributed multi-robot patrol: A scalable and fault-tolerant framework. *Robotics and Autonomous Systems*, 61(12):1572 – 1587.

Santana, H., Ramalho, G., Corruble, V., and Ratitch, B. (2004). Multi-agent patrolling with reinforcement learning. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, 2004. AAMAS 2004.*, pages 1122–1129.

Sugiyama, A., Sea, V., and Sugawara, T. (2016). Effective task allocation by enhancing divisional cooperation in multi-agent continuous patrolling tasks. In *2016 IEEE 28th Int. Conf. on Tools with Artificial Intelligence (ICTAI)*, pages 33–40.

Sugiyama, A. and Sugawara, T. (2015). Meta-strategy for cooperative tasks with learning of environments in multi-agent continuous tasks. In *Proc. of the 30th Annual ACM Symposium on Applied Computing*, SAC '15, pages 494–500, New York. ACM.

Wolf, D.F., S. G. (2005). Mobile robot simultaneous localization and mapping in dynamic environments. In *Auton Robot*, volume 19, page 53–6.

Wooldridge, M. (2009). *An Introduction to MultiAgent Systems*. Wiley Publishing, 2nd edition.

Xie, J. and Liu, C.-C. (2017). Multi-agent systems and their applications. *Journal of International Council on Electrical Engineering*, 7(1):188–197.

Yoneda, K., Sugiyama, A., Kato, C., and Sugawara, T. (2015). Learning and relearning of target decision strategies in continuous coordinated cleaning tasks with shallow coordination. *Web Intelligence*, 13(4):279–294.

Yongguo Mei, Yung-Hsiang Lu, Lee, C. S. G., and Hu, Y. C. (2006). Energy-efficient mobile robot exploration. In *Proc. 2006 IEEE Int. Conf. on Robotics and Automation (ICRA 2006)*, pages 505–511.

8 List of Publications

- **Lingying Wu** and Toshiharu Sugawara (2019). Strategies for Energy-Aware Multi-Agent Continuous Cooperative Patrolling Problems subject to Requirements. In *Proceedings of the 22nd International Conference on Principles and Practice of Multi-Agent Systems (PRIMA 2019)*, Lecture Notes in Artificial Intelligence (LNAI), volume 11873, page 585-593, Springer.

- **Lingying Wu**, Ayumi Sugiyama and Toshiharu Sugawara (2019). Energy-Efficient Strategies for Multi-Agent Continuous Cooperative Patrolling Problems. In *Proceedings of 23rd International Conference on Knowledge-Based and Intelligent Information and Engineering Systems (KES 2019)*, Procedia Computer Science, volume 159, page 465-474, Elsevier.