Journal of Biosystems Engineering
https://doi.org/10.1007/s42853-022-00157-7

Online ISSN 2234-1862
Print ISSN 1738-1266

ORIGINAL ARTICLE

# Performance Evaluation of Path Planning and Coordination Algorithms for Multiple UGVs in Smart Farm

Yuseung Jo[1,2] · Hyoung Il Son[1,2]

Received: 6 May 2022 / Revised: 1 September 2022 / Accepted: 5 September 2022


## Abstract

**Purpose** Path planning and coordination algorithms are needed for deploying multiple unmanned ground vehicles (UGVs) on a smart farm. These algorithms differ in performance because of constraints such as the number of robots, obstacle configuration, and environment density. Therefore, this paper aims to select the most robust path planning and coordination algorithm by evaluation under the constraints of smart farms.

**Methods** In this study, we evaluated three path planning and coordination algorithms: (1) conflict-based search (CBS), (2) enhanced CBS (ECBS), and (3) push and rotate (PAR) algorithms. The evaluation was performed in an environment mimicking an actual smart farm. We obtained the number of collisions, success rates, and total travel distance (TTD) data. For statistical analysis, a t-test was performed on TTD data.

**Results** The statistical analysis of the TTD confirmed no difference between the performance of the three algorithms. The ECBS and PAR algorithms exhibited the same 100% success rate for robot systems with fewer than 10 robots. Furthermore, the PAR algorithm exhibited a robust performance in all robot systems with at most 20 robots.

**Conclusions** We evaluated three conventional path planning and coordination algorithms for a multi-UGV system. The evaluation was performed in an environment that mimicked an actual smart farm. Although there is no difference in the path generation performance of all algorithms, the results revealed that the PAR algorithm was the most robust given the constraints in a smart farm.

**Keywords** Coordination · Harvest automation · Multi-UGV · Multi-agent path finding · Path planning

## Introduction

Smart farms are being actively distributed on a large scale. Moreover, farmers need a large amount of labor to manage large-scale smart farms. Intensive agricultural labor is required during the fruit vegetable harvesting and growth periods. Controlling the growth-affecting parameters (e.g., illumination, temperature, humidity, and $CO_2$ level) to near-optimal has long been a challenge for agriculture. During the growth period of large-scale smart farms, excessive labor is often used to maintain optimal conditions for fruits and vegetables. Given this need, research targeting the growth period is ongoing. One representative study aimed to maintain optimal conditions for growth by collecting and analyzing agricultural data with Internet of Things(IoT)–based smart farm systems, reaching a commercialized level (Muangprathub et al., 2019; Kim et al., 2020; Majumdar et al., 2021).

Agricultural tasks during the harvesting period include labor-intensive tasks such as harvesting and transporting fruits and vegetables. Therefore, automation during this period aims primarily at the development of harvesting and transporting robots (Zhao et al., 2011; Peng & Vougioukas, 2020; Kim et al., 2020; Gao et al., 2022). As an approach to agricultural robots during the harvest period, unmanned ground vehicles (UGVs) are often chosen because of their advantages in transportability, runtime, and stability for extending to transporting robots and mobile manipulators which can replace human labor-intensive tasks. However, studies on UGV-based

✉ Hyoung Il Son
hison@jnu.ac.kr

[1] Department of Convergence Biosystems Engineering, Chonnam National University, 77 Yongbong-ro, Buk-gu, Gwangju 61186, Republic of Korea

[2] Interdisciplinary Program in IT-Bio Convergence System, Chonnam National University, 77 Yongbong-ro, Buk-gu, Gwangju 61186, Republic of Korea

Published online: 15 September 2022

systems have been conducted using a single robot, which does not improve the current low work efficiency of agricultural tasks in smart farms.

In this paper, we assume a multi-robot system-based approach to improve work efficiency. We use UGVs for agricultural tasks in a smart farm, by impersonating human workers, who perform the harvesting work in collaboration (Ju & Son, 2021a). The actual harvesting work is collaboratively performed by the workers in charge of harvesting and transportation. Therefore, harvesting can be considered a tightly coordinated task that demands inter-robot cooperation. Furthermore, in this context, it is necessary to use multiple UGVs to improve the efficiency of the autonomous harvesting system.

When using multiple UGVs, unlike a single robot, considering additional constraints (e. g., path planning) is critical (Kim & Son, 2020). Multiple UGVs should avoid environments that contain static obstacles, as with a single robot. Moreover, collisions should be avoided through coordination with other UGVs, which are dynamic obstacles (Švestka & Overmars, 1998). This study uses path planning and coordination algorithms to avoid static and dynamic obstacles. These algorithms have unique performance because of constraints such as the number of robots, obstacle configuration, and environment density. Therefore, for successful adoption of path planning and coordination control algorithms, an evaluation should be performed considering constraints (e.g., movement scenarios of UGVs in a smart farm, obstacle configuration, number of robots, and environment density) (Stern et al., 2019).

Multi-agent path finding (MAPF) is a traditional approach to multi-robot path planning and coordination. It addresses the problem defined as finding a collision-free path for $k$ agent $a_1 \ldots a_k$ to the initial vertex and the desired vertex in an undirected graph $G(V,E)$ where the environment is mapped. In MAPF problems, each agent occupies one vertex, and a collision is defined when two or more agents desire to occupy one vertex or edge. All agents can perform one action (i.e., *move* and *wait*) per discretized time step, and the cost of the action is 1. The *move* action targets the adjacent vertex. The MAPF problem aims to minimize the sum of the costs of all agents (Felner et al., 2017).

MAPF solution algorithms have three representative solutions: search-based solution, rule-based solution, and hybrid solution that combines the two aforementioned solutions. The search-based and rule-based solutions have wide research scalability and high reliability; therefore, they have been the focus of several studies. The conflict-based search (CBS) and enhanced CBS (ECBS) algorithms, which return representative search-based solutions, and the push and rotate (PAR) algorithm, which provides rule-based solutions, were selected as the targets for comparison and evaluation. Each algorithm is proposed for a different detailed objective based on the objective of the MAPF algorithm.

For example, CBS is an algorithm that calculates the optimal path, and a large number of agents leads to an exponential increase in computational resources. The ECBS algorithm was proposed to solve this problem by finding sub-optimal paths. As illustrated in the example, the performance of the algorithm varies depending on the target environment, the number of agents, and the density of obstacles. We cannot be confident of the reliability of rule-based, search-based algorithms. Accordingly, we aim to determine which algorithms (i.e., CBS, ECBS, and PAR which are representative rule-based and search-based algorithms) are most robust in a smart farm.

## Related Works

Artificial intelligence has reached a practical level with the advancement of graphical processing units. Multi-robot systems have many potential conditions to consider, unlike single robots that avoid static obstacles with optimal paths. Reinforcement learning is suitable for complex problems such as MAPF. Efforts to apply reinforcement learning to multi-robot path planning and coordination (beyond single robots) have been significant areas of study in the last decade (Damani et al., 2021; Huang et al., 2021; Liu et al., 2021). The Q-network-based and the actor-critic-based methods achieved a large scientific spreading for multi-robot control (Canese et al., 2021). Accordingly, the path planning and coordination problem in multi-robot systems is treated as multi-agent reinforcement learning (Zhang et al., 2021). Reinforcement learning-based approaches facilitate selecting the correct parameters for the algorithm to apply to the environment. However, an empirical approach is essential, and there is a limit to the exponential increase in learning time as the system grows.

One study used discrete event systems (DESs) as an alternative approach to controlling multiple agricultural robots to efficiently perform agricultural tasks such as tributary mapping, modeling the system as automata-based discrete events, and designing a supervisory controller (Ju & Son, 2021b; Seol et al., 2021). Because of the advantages of the DES approach, a hierarchical approach is possible by implementing a modular form of the approach. However, DES-based approaches require learning new languages, the semantics of which are ambiguous during the modeling process.

### Objective and Contribution

In this study, we select path planning and coordination algorithms suitable for smart farms to apply multi-UGV systems. We evaluate the three typical algorithms (i.e., CBS, ECBS, and PAR) in an experimental environment that mimics smart farms.

## Path Planning and Coordination of a Multi-UGV System

### Smart Farm Environment

Most smart farms have general and common characteristics. A smart farm is divided primarily into three categories: (1) a greenhouse for crop growth, (2) a sorting workspace for selecting by marketability like weight, surface damage, and color, and (3) a warehouse that manages overall agricultural equipment and materials. Even exceptional smart farms, do not have a configuration that deviates significantly from these categories. Figure 1 illustrates several common characteristics of the smart farm and the environment in which the multi-robot system will be applied. As depicted in Fig. 1a, b, gutters for fruit and vegetable cultivation are arranged in parallel in the greenhouse.

Furthermore, hot water pipes for temperature control in the smart farm are arranged in parallel between the gutters. The hot water pipes serve as the track along which the UGVs move the width between the two gutters can accommodate one UGV. The corridor in the greenhouse in Fig. 1b allows two UGVs to travel in parallel. As depicted in Fig. 1c, the pedestrian path for personnel in a smart farm was not covered by the UGV guideline for safety reasons. Therefore, the pedestrian path can be treated as a static obstacle because the robot cannot cross it.

Smart farms include three agricultural tasks: harvesting, transportation, and sorting. Furthermore, as depicted in Fig. 2, the workspace is divided into (1) a warehouse, (2) a sorting workspace, and (3) a harvesting workspace. The transportation and harvesting tasks are suitable for automation using UGVs. However, the sorting task is performed by an individual sorting machine and excluded in our scenario because it is not considered by the multi-robot system.

### Multiple Harvesting and Transporting Robot

This study assumed a situation in which harvesting and transporting UGVs are used in a smart farm to perform harvesting and transportation tasks. For harvesting tasks based on multiple UGVs, the autonomous harvesting system is divided into harvesting robots, which are mobile manipulators, and transporting robots, which are UGVs equipped with transport modules. Harvesting robots are assigned only harvesting tasks.

Human harvesting workers exist only in the warehouse and harvesting workspace. Therefore, the harvesting robots travel only between the harvesting workspace and warehouse, as depicted in Fig. 2. The transporting robots are assigned the transporting task between the three workspaces. All robots move when a new task is assigned and when they must leave their designated position because of a low battery level, functional failure, or other reasons. The robots follow the paths obtained using the MAPF algorithm in coordination with other robots to avoid static and dynamic obstacles. We describe the three MAPF algorithms selected in this study in the "Multi-Agent Path Finding Algorithm" section.

## Multi-Agent Path Finding Algorithm

Multi-agent path finding (MAPF) is a traditional problem for multi-robot systems. A major constraint is that each robot must be able to follow a path without colliding with another robot. During robot movement, the agent, which in this case is the robot, can move to an adjacent location or wait at the current location (Stern et al., 2019). A collision is a situation in which two or more robots simultaneously occupy the same position. All agents must avoid conflict and reach their respective destinations. A path is acquired using the MAPF algorithm to avoid static and dynamic obstacles. This section describes the representative search-based and rule-based algorithms selected for comparison and evaluation in this study.

### Conflict-Based Search Algorithm

Conflict-based search (CBS) algorithm is a representative search-based and constraint tree (CT)-–based algorithm (Sharon et al., 2015; Hönig et al., 2019). The searching is performed by creating one child node from two parent nodes. In searching, the CBS algorithm performs research that considers the collision period if a conflict occurs between the parent node combinations. For example, if $a_i$ and $a_j$ collide in period $t$ in vertex $v$, the constraint ($a_i$, $v$, $t$) is added to $a_i$ or $a_j$ (in this example, $a_i$) the constraint prevents $a_i$ from


(a) (b) (c)

**Fig. 1** Typical characteristics of a smart farm environment: **a** harvest area of harvesting workspace, **b** harvesting workspace, and **c** sorting workspace

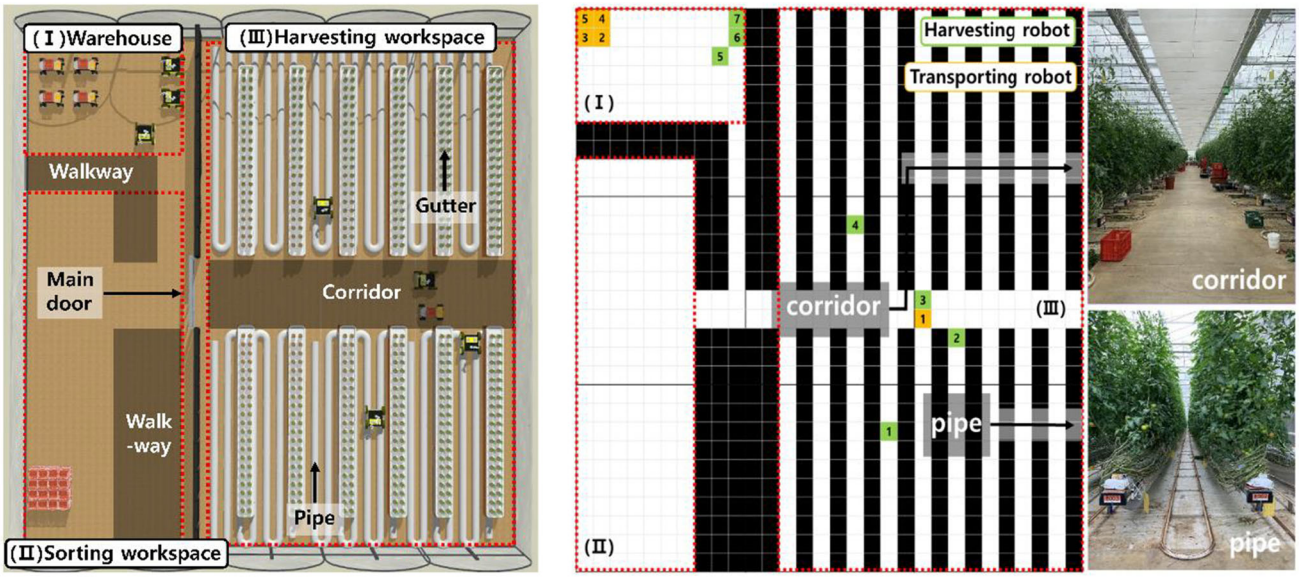


**Fig. 2** Multi-UGV system in smart farm assumed in this study: top-view (**a**) grid (**b**) expression and actual smart farm

moving to vertex $v$ in period $t$. The parent nodes start with the optimal path for a single robot and are combined until the total travel distance (TTD) is minimal for the multi-robot system. The path calculated through the CBS algorithm is the optimal path for multiple robots.

The efficiency of the overall task and optimality of the path are directly related. Among the MAPF algorithms, the CBS algorithm that can derive the optimal path was selected for comparison and evaluation. The CBS pseudo-code is presented in Table 1 (Sharon et al., 2015).

## Enhanced CBS Algorithm

Solving the MAPF problem is not limited to finding the optimal path for all robots. Calculating the optimal path may be time-intensive depending on the size of multi-robot systems. Therefore, quickly deriving a suboptimal path is another representative approach for solving MAPF problems. The suboptimal path is derived through a focal search considering the suboptimality of the algorithm parameter (Pearl & Kim, 1982). The enhanced CBS (ECBS) algorithm is a CT-based search-based algorithm (Barer et al., 2014). However, the ECBS algorithm was proposed to improve the excessive computation for congestion situations of the CBS algorithm. For example, using multiple robots in a smart farm can require extensive time to calculate the optimal path depending on the density of obstacles and the number of robots. Planning and following the suboptimal path can be more efficient in these scenarios than slowly exploring the optimal path. Therefore, the ECBS algorithm is selected for comparison. The pseudo-code of ECBS is presented in Table 2 (Huang et al., 2021).

## Push and Rotate Algorithm

Returning the moved robot to its destination while avoiding obstacles is essential in the multi-robot system. The PAR algorithm solves this problem by choosing operations based on

**Table 1** Pseudo-code for CBS

**Algorithm**: conflict-based search

**Input**: MAPF instance
**1:** *Root*.constraints=∅
**2:** *Root*.solution = find individual path by the low level()
**3:** *Root*.cost = Sum of individual cost of root.solution
**4:** insert *Root* to OPEN // OPEN = empty set
**5:** **while** OPEN *not empty* **do**
**6:** $P$←best node from OPEN // lowest cost node
**7:** Validate the paths in $P$ until a conflict occurs
**8:** **if** $P$ has no conflict **then**
**9:** **return** $P$. solution// $P$ is goal
**10:** $C$←first conflict $(a_i, a_j, v, t)$ in $P$
**11:** **foreach** agent $a_i$ in $C$ **do**
**12:** $A$←new node
**13:** $A$. constraint←$P$. constraint + $(a_i, v, t)$// new node $A$ considered position $v$ and time $t$ of collision
**14:** $A$. solution←$P$. solution
**15:** Update $A$. *solution* by involking low level $(a_i)$
**16:** $A$. *cost*=Sum of Individual cost of $A$. solution
**17:** **if** A. cost<∞ // $A$. solution was found **then**
**18:** Insert $A$ to OPEN

**Table 2** Pseudo-code for ECBS

**Algorithm:** enhanced conflict–based search

**Input**: MAPF instance and suboptimality factor $w$

**1:** Generate the root CT node $R$ with an initial solution // same with CBS 1-3

**2:** Initialize OPEN ← $\{R\}$

**3:** $LB \leftarrow R_{LB}$, and and initialize focal list ← $\{R\}$ // $LB$ = lower bound on the optimal solution of the entire problem,

// $R_{LB}$= lower bound of CT node $R$

**4: while** OPEN not empty **do**

**5:** $N \leftarrow$ a CT node with the minimum heuristic component in

**6:** **if** $N$ has no conflict **then**

**7:** **return** $N_{sol}$// compare two CT nodes

**8:** Delete $N$ from the open and focal lists

**9:** **if** $min_{N \in \text{OPEN}} N_{LB}$ **then**

**10:** $LB \leftarrow min_{N \in \text{OPEN}} N_{LB}$ //update $LB$

**11:** $\leftarrow N \in \text{OPEN} : N_{LB} \leq wLB$

**12:** Pick a conflict in $N_{conf}$

**13:** Generate 2 child CT nodes $N^1$ and $N^2$ of $N$

**14:** Call low-level search for $N^i$ to calculate $N^i_{sol}$, $N^i_{\text{cost}}$, and $N^i_{\text{conf}}$ for $i=1, 2$

**15:** Add $N^i$ to OPEN for $i=1, 2$

**16:** Add $N^i$ to if $N^i_{\text{cost}} \leq wLB$ for $i=1, 2$

**17: Return** No solution

the priority of all robots. Of the rule-based algorithms, the representative push and rotate (PAR) algorithm determines whether to push or rotate other robots according to their priority. As an example of a push operation, we can assume that the lower priority agents $a_{\text{priority=2}}$ and $a_{\text{priority=3}}$ arrived at their desired position and blocked the corridor in Fig. 2b. The high-priority agent $a_{\text{priority=1}}$ follows its path by pushing agent $a_{\text{priority=2}}$ or $a_{\text{priority=3}}$ into adjacent positions and the moved robot returns to its destination. In this time, $a_{\text{priority=2}}$ can also be pushed out $a_{\text{priority=3}}$. For a rotation operation, all robots corresponding to the collision situation are assumed to be in a circular formation. After the collision resolution, the circle is rotated to return the moving robot to its original position. PAR algorithms tend to cause low-priority agents to choose a *wait* action when a high-priority robot is following a path. However, this tendency is not applicable when two paths do not overlap.

Similar to that derived by ECBS, the derived path is sub-optimal. However, the PAR algorithms are selected because, unlike ECBS, they use additional operations to resolve efficient collision situations. Each detailed operation of the PAR algorithm can be identified by referring to the study by Wilde et al. (2014). The solved algorithm used in the PAR algorithm is presented in Table 3.

## Experimental Method

### Experimental Environment

Simulation-based experiments were performed to evaluate path planning and coordination algorithms in a smart farm. We calculated the paths using MAPF algorithms. Moreover, the generated paths were visualized in the simulation. For a proper performance evaluation, the simulation environment was implemented as depicted in Fig. 2b. In addition to static obstacles equivalent to environments where robots cannot pass we also treated the walkways in Fig. 1c as additional obstacles. The environment and robots are simplified into a grid-cell structure in the simulation. From this perspective, the robot appears to occupy one cell and is expressed as a dot. Because the robot is expressed as a dot, a collision caused by the robots' bodies is not considered in this simulation. However, the collision was assumed when two robots desired to occupy one cell simultaneously. The experimental environment and algorithms were implemented using C++ and Python in 64-bit Linux environments with an Intel Core CPU i3-8350K@4.00 GHz with 16.0 GB RAM.

**Table 3** Pseudo-code for PAR

**Algorithm 3**: Pseudo-code for solve of PAR

```
Initialize the sequence of moves and paths of resolving agent to the empty list // resolving agent is the agent corresponding to a collision instance
The set of finished agents ←∅
if input graph is polygon then
    return is-polygon = true
while the set of finished agent ≠ the set of agents do
    if no agent selected then
        Select next priority agent
    if is-polygon then
        Choose the shortest path // swap is impossible
    while position of selected agent ≠ destination do
        Move one step closer to destination
        if path of selected agent is on path of resolving agents then
            Rotate operation(resolving agents)
        elif rotation operation = false
            Push operation
        else swap operation
    The set of finished agent← selected agent
    Initialize path of resolving agent
    while any agents need to be returned to their goal location after operation do
        check whether their goal location is occupied
        if their goal location is unoccupied then
            move agent to goal location
        else break inner loop, continue outer Loop
return the sequence of moves
```

## Experiment Scenarios

This study aimed to evaluate the performance of the MAPF algorithm. We assumed that each robot plans its path and cooperates with others as it moves to a given destination. A scenario in which multiple mobile manipulators and transporting robots exist was adopted for the experiment. Each robot acquired a path through a MAPF algorithm; then, the planned path was driven along based on the guideline. The transporting robot moved freely through the harvesting and sorting workspace and lifted the post-harvesting products to the sorting space, while harvesting robots were active primarily in the harvesting workspace. Detailed movement scenarios are presented in Table 4.

The experimental environment was constructed by mimicking a part of the actual smart farm environment. In the experimental environment, as the robot travels along the buried guideline, any region other than the guideline is considered an obstacle. The suboptimality of the ECBS algorithm was set to 1.3 for the experiment. Moreover, the cells to the top, bottom, left, and right of those occupied by the robots are considered adjacent to the robots.

**Table 4** Index of inter-workspace movement scenarios

| Start position | Desired position | Transporting robot Movement scenario | Harvesting robot |
|---|---|---|---|
| **I** | **II** | O | X |
| | | Robot replacement | |
| **I** | **III** | O | O |
| | | Task start /robot replacement | |
| **II** | **I** | O | X |
| | | Task complete / robot replacement | |
| **II** | **III** | O | X |
| | | Start transportation | |
| **III** | **I** | O | O |
| | | Task complete / robot replacement | |
| **III** | **II** | O | X |
| | | Transportation | |

Figure 3 illustrates the initial and desired points of each robot and the paths of seven robots acquired by (a) CBS, (b) ECBS, and (c) PAR algorithms in their respective colors. All movements of robots are based on the movement scenario in Table 4. The circle symbol indicates that the robot has chosen the *wait* action, illustrated in Fig. 3b, c. However, unlike ECBS, the circle symbol in PAR indicates that choosing the *wait* action for a path of a higher priority robot does not overlap the lower priority robot.

For quantitative evaluation of the algorithm, the number of robots was changed to 10, 15, and 20, and the experiments were repeated 20 times for the three algorithms. The starting point and destination of the robot were randomly assigned, and none of the points of each robot overlapped.

## Performance Metrics

The performance metrics for evaluation were selected as presented in Table 5. The number of collisions and TTD are direct measures of how efficiently multiple robot paths were planned. $X_C$ is the number of collisions in one case, $X_T$ is the number of movements in the Manhattan distance made by all robots in one case for a particular TTD, $N_s$ is the number of successful planning, and $N$ is the total number of path planning ($N = 20$). For the success rate in this study, failure was defined as planning that required more than 5 min, an outlier that is usually outside the path planning time (Sharon et al., 2015; Barer et al., 2014; Sharon et al., 2012; Huang et al., 2021).

# Experimental Results

The results of the experiments on path planning and coordination for systems containing 10, 15, and 20 robots using the

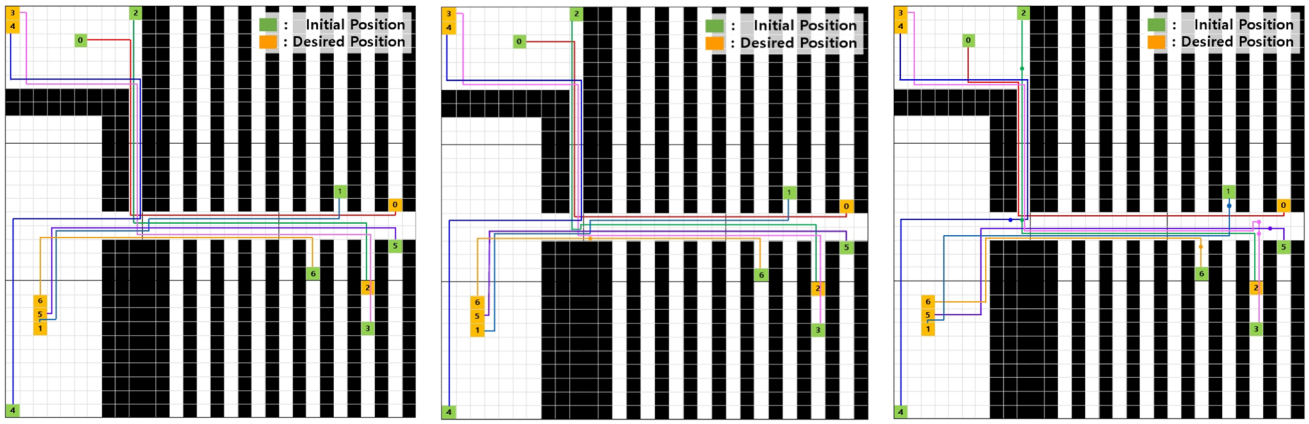


**Fig. 3** Paths with initial and desired positions of the seven robots generated by each algorithm: CBS (**a**), ECBS (**b**), and PAR (**c**) algorithms. Each path has a respective color. The circle symbols indicate that the robot has chosen a wait action at that location

CBS, ECBS, and PAR algorithms are presented in Table 6. The number of collisions, an essential element of a multi-robot system, was confirmed as 0 for all the experiments. The TTD was calculated for cases corresponding to successful path planning. For example, for a 10-robot system, the TTD of the CBS algorithm was derived in 14 out of 20 cases. The asterisk symbol was used to identify the algorithm that exhibited the highest success rate for each number of robots.

## Statistical Analysis and Result

The path planning performance of each algorithm can be directly confirmed through the TTD: the lower the TTD, the higher the path planning performance. The Shapiro–Wilk, and Kolmogorov–Smirnov tests were performed to quantitively evaluate the TTD, confirming that the sample satisfied the criteria for normality. The sample satisfied the criteria for normality, so a t-test ( a statistical test) was performed to test the hypothesis. In this study, we selected a 95% confidence interval as the significance level. The null hypothesis "there is no difference in path planning performance between the two algorithms" was rejected when the p-value was less than 0.05.

Each case of 20 iterations has an individual starting point and destination. For meaningful statistical analysis, new TTD data based on the same starting point and destination are required, as presented in Table 7. We deleted the failure case for the new TTD data of CBS, ECBS, and PAR algorithms. For example, in the system with 10 robots, the success rate of the CBS algorithm was confirmed at 70% for 20 iterations. We deleted the six failure cases. Moreover, the three algorithms can be compared using 14 cases on the same basis, indicating the same starting points and destinations.

The newly derived TTD was 259.14 cells with CBS, 283.2 cells with ECBS, and 288 cells with PAR. For systems with 15 and 20 robots, CBS had success rates of 5% and 0%. The sample was insufficient for comparison and exhibited low performance based on the designated performance indicators; therefore, it was excluded from the comparison. We confirmed that the $p$ value of the comparing of the CBS, ECBS, and PAR algorithms in systems with 10, 15, and 20 robots was higher than 0.05, which had no statistical significance in path planning.

**Table 5** Performance metric

| Task | Item | Equation |
|---|---|---|
| Navigation | Collision | $1/N_s \cdot \sum X_C$ |
| Path planning | Total travel distance | $1/N_s \cdot \sum X_T$ |
| | Success rate | $N_S/N \cdot 100$ |

## Quantitative Evaluation

The performance of the MAPF algorithm can also be determined based on the success rate based on the calculation time (Wilde et al., 2014). In the system with 10 robots, the success rates of the CBS, ECBS, and PAR algorithms were 70%, 100%, and 100%. In the system with 15 robots, the rates were 5%, 80%, and 100%. Furthermore, for the multi-robot systems with more than 20 robots, the rates were 0%, 55%, and 100% for the CBS, ECBS, and PAR algorithms.

The experimental results reveal that the low success rate of the CBS algorithm made the algorithm unsuitable for all situations. When using multiple harvesting and transportation robots in an environment simulating a smart farm, the ECBS and PAR algorithms exhibited a success rate of 100%. Furthermore, these two algorithms achieved adequate performance in robot systems with fewer than 10 robots. Furthermore, the PAR algorithm exhibited higher performance in multi-robot systems with at most 20 robots.

**Table 6** Experiment results

| # of robots | Metric | CBS | ECBS | PAR |
|---|---|---|---|---|
| 10 robots | TTD (cells) | 259.14 | 292.55 | 290.3 |
| | Success rate | 70% (14/20) | 100% (20/20)* | 100% (20/20) * |
| | Collision | 0 | 0 | 0 |
| 15 robots | TTD (cells) | 359 | 457.88 | 452.4 |
| | Success rate | 5% (1/20) | 80% (16/20) | 100% (20/20) * |
| | Collision | 0 | 0 | 0 |
| 20 robots | TTD (cells) | **NaN** | 603 | 606.4 |
| | Success rate | 0% (0/20) | 55% (11/20) | 100% (20/20) * |
| | Collision | **NaN** | 0 | 0 |

## Discussion

### Need for Multiple Robots

Using multiple UGVs in a smart farm can reduce fruit harvesting time. Robotic cooperation can also be achieved, making the use of multiple UGVs an efficient approach. Fruits grow atypically and are planted densely. Therefore, a single harvesting robot would require significant time to harvest each fruit because of its low perception rate and the difficulty in motion planning of the manipulator. Because of the limitations of these harvesting robots, the operation of a single robot is insufficient for actual harvesting tasks. However, if multiple robots that collaborate like human workers are used instead of a single harvesting robot, the efficiency of automatic harvesting tasks can be enhanced.

### Need for Evaluation

The suitability and efficiency of the MAPF algorithm are usually determined indirectly through benchmark results in a comprehensive and unclassified environment. The conventional evaluation of the MAPF algorithm might not be appropriate for all environments, and its use would result in inaccurate results. Furthermore, a small number of large obstacle densities are assumed for the conventional evaluation. Smart farms do not have much space allocated to fruit stems for optimal production, and this environment is dense. However, unlike smart farms the environment assumed during evaluation, the MAPF algorithm has sufficient space allocated to agents; consequently, the algorithm is free to select the optimal path for each robot. Therefore, a dense environment such as a smart farm should require evaluations that differ from conventional evaluations. When evaluated according to the smart farm environment and work scenarios, appropriate references for using multiple robots in a smart farm can be acquired.

**Table 7** Re-derived TTD data and statistical analysis results

| | Algorithm<br>Total travel distance | *p* value |
|---|---|---|
| 10 robots | CBS ↔ ECBS<br>259.14 283.21 | 0.26 |
| | CBS ↔ PAR<br>259.14 288.00 | 0.20 |
| | ECBS ↔ PAR<br>283.21 288.00 | 0.89 |
| 15 robots | ECBS ↔ PAR<br>457.88 453.18 | 0.81 |
| 20 robots | ECBS ↔ PAR<br>603.00 599.64 | 0.91 |

### Success Rate–based Evaluation

The MAPF algorithm performs a search at the central controller and assigns the calculated path to each robot. For practical usage of multi-robot systems with the MAPF algorithm, a limitation of the central control method is that all robots must be re-planned for the path planning of one robot. Therefore, this study assumes that algorithms with high success rates are robust when all robots are re-planned.

For the case of the success rate, the CBS algorithm and the ECBS algorithm exhibited variable results for the different numbers of robots. These two algorithms are CT-based exploration algorithms that consider all robot cases to address the collision problem during computation. Consequently, the number of robots to be compared increases exponentially. In contrast, because of robot priority, the PAR algorithm can handle collision scenarios more precisely with relatively few operations compared with the CBS and ECBS algorithms assuming the same priority.

## Conclusion

This study quantitatively compared and evaluated the performance of path planning and coordination algorithms for application to multiple robot systems within smart farms. The CBS, ECBS, and PAR algorithms representative MAPF algorithms, were selected, and their performance was compared. The smart

farm was subdivided into three workspaces for an evaluation suitable for multi-robot systems in a smart farm. Furthermore, we designed scenarios for movement between workspaces to perform iterative experiments and quantitative evaluations.

The results revealed that there was no generated path difference for all algorithms. However, for the success rate, the ECBS and PAR algorithms performed appropriately in systems with fewer than 10 robots, and the PAR algorithm was robust in smart farm environments with multi-robot systems with fewer than 20 robots.

**Acknowledgements** This research was supported in part by the Korea Institute of Planning and Evaluation for Technology in Food, Agriculture, and Forestry (IPET) through the Smart Farm Innovation Technology Development Program, funded by the Ministry of Agriculture, Food and Rural Affairs (MAFRA), under Grant 421031-04.

## Statements and Declaration

**Competing Interests** The authors declare no competing interests.

## References

Barer, M., Sharon, G., Stern, R., & Felner, A. (2014). Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem. *Proc. 7th Annu. Symp. Combinatorial Search*, 19-27.

Canese, L., Cardarilli, G. C., Nunzio, L. D., Fazzolari, R., Giardino, D., Re, M., & Spanò, S. (2021). Multi-agent reinforcement learning: A review of challenges and applications. *Applied Sciences, 11*(11), 4948. https://doi.org/10.3390/app11114948

Damani, M., Luo, Z., Wenzel, E., & Sartoretti, G. (2021). PRIMAL2: Pathfinding via reinforcement and imitation multi-agent learning-lifelong. *IEEE Robotics and Automation Letters, 6*(2), 2666–2673. https://doi.org/10.1109/LRA.2021.3062803

Felner, A., Stern, R., Shimony, S. E., Boyarski, E., Goldenberg, M., Sharon, G., Sturtevant, N. R., Wagner, G., & Surynek, P. (2017). Search-based optimal solvers for the multi-agent pathfinding problem: Summary and challenges. *The 10th International Symposium on Combinatorial Search, 8*(1), 29–37.

Gao, J., Zhang, F., Zhang, J., Yuan, T., Yin, J., Guo, H., & Yang, C. (2022). Development and evaluation of a pneumatic finger-like end-effector for cherry tomato harvesting robot in greenhouse. *Computers and Electronics in Agriculture, 197*, 106879. https://doi.org/10.1016/j.compag.2022.106879

Hönig, W., Kiesel, S., Tinka, A., Durham, J. W., & Ayanian, N. (2019). Persistent and robust execution of MAPF schedules in warehouses. *IEEE Robotics and Automation Letters, 4*(1), 1125–1131. https://doi.org/10.1109/LRA.2019.2894217

Huang, Taoan, Dilkina, B., & Koenig, S., (2021). Learning node-selection strategies in bounded suboptimal conflict-based search for multi-agent path finding. *International Joint Conference on Autonomous Agents and Multiagent Systems*.

Ju, C., & Son, H. I. (2021a). A hybrid systems-based hierarchical control architecture for heterogeneous field robot teams. *IEEE Transactions on Cybernetics*, 1–14. https://doi.org/10.1109/LRA.2019.2941178.

Ju, C., & Son, H. I. (2021b). Modeling and control of heterogeneous field robots under partial observation. *Information Sciences, 580*(1), 419–435. https://doi.org/10.1016/j.ins.2021.08.071

Kim, J., & Son, H. I. (2020). A voronoi diagram-based workspace partition for weak cooperation of multi-robot system in orchard. *IEEE Access, 8*(1), 20676–20686. https://doi.org/10.1109/ACCESS.2020.2969449

Kim, W. S., Lee, W. S., & Kim, Y. J. (2020). A review of the applications of the Internet of Things (IoT) for agricultural automation. *Journal of Biosystems Engineering, 45*, 385–400. https://doi.org/10.1007/s42853-020-00078-3

Liu, Z., Liu, Q., Tang, L., Jin, K., Wang, H., Liu, M., & Wang, H. (2021). Visuomotor reinforcement learning for multirobot cooperative navigation, IEEE transactions on automation science and engineering, Early Access, 1-12. https://doi.org/10.1109/TASE.2021.3114327.

Majumdar, P., Mitra, S., & Bhattacharya, D. (2021). IoT for promoting Agriculture 4.0: A review from the perspective of weather monitoring, yield prediction, security of WSN protocols, and hardware cost analysis. *Journal of Biosystems Engineering, 46*, 440–461. https://doi.org/10.1007/s42853-021-00118-6

Muangprathub, J., Boonnam, N., Kajornkasirat, S., Lekbangpong, N., Wanichsombat, A., & Nillaor, P. (2019). IoT and agriculture data analysis for smart farm. *Computers and Electronics in Agriculture, 156*(1), 467–474. https://doi.org/10.1016/j.compag.2018.12.011

Pearl, J., & Kim, J. H. (1982). Studies in semi-admissible heuristics. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 4*(1), 392–399. https://doi.org/10.1109/TPAMI.1982.4767270

Peng, C., & Vougioukas, S. G. (2020). Deterministic predictive dynamic scheduling for crop-transport co-robots acting as harvesting aids. *Computers and Electronics in Agriculture, 178*, 105702. https://doi.org/10.1016/j.compag.2020.105702

Seol, J., Ju, C., & Son, H. I. (2021). A leader-follower control of multi-UAV for tributary mapping based on supervisory control theory: A preliminary result. *Institute of Control, Robotics and Systems, 2021*(1), 615–616.

Sharon, G., Stern, R., Felner, A., & Sturtevant, N. (2012). Meta-agent conflict-based search for optimal multi-agent path finding. *Proc. 5th Annual Symposium on Combinatorial Search*, 97-104.

Sharon, G., Stern, R., Felner, A., & Sturtevant, N. R. (2015). Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence, 219*(1), 40–66. https://doi.org/10.1016/j.artint.2014.11.006

Stern, R., Sturtevant, N., Felner, A., Koenig, S., Ma, H., Walker, T., Li, J., Atzmon, D., Cohen, L., Kumar, T. K. S., Boyarski, E., & Bartak, R. (2019). Multi-Agent Pathfinding: Definitions, Variants, and Benchmarks. *Proc. 12th Annu. Symp. Combinatorial Search,* 151-158.

Švestka, P., & Overmars, M. H. (1998). Coordinated path planning for multiple robots. *Robotics and Autonomous Systems, 23*(1), 125–152. https://doi.org/10.1016/S0921-8890(97)00033-X

Wilde, B. d., Mors, A. W., & Witteveen, C. (2014). Push and rotate: A complete multi-agent pathfinding algorithm. *Journal of Artificial Intelligence Research, 51*(1), 443–492. https://doi.org/10.1613/jair.4447

Zhang, K., Yang, Z., & Başar, T. (2021). Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of Reinforcement Learning and Control*, 325, 321-384. https://doi.org/10.1007/978-3-030-60990-0_12

Zhao, D. A., Jidong, L., Wei, J., Ying, Z., & Yu, C. (2011). Design and control of an apple harvesting robot. *Biosystems Engineering, 110*(2), 112–122. https://doi.org/10.1016/j.biosystemseng.2011.07.005